NexSRS™ Peering Server

User Manual V.3.0
Linux version

NexSRS™ Peering Server

© TransNexus, Inc.
555 Fourteenth Street
Atlanta • Georgia • 30318
Phone 404-526-6060
Support@TransNexus.com

NexSRS Peering Server Version 3.0.4
for Linux
27 April 2007

The information herein is the property of TransNexus, Inc. It is not to be distributed by the recipient to third parties without prior written permission from TransNexus, Inc. Copyright 2001-2007. All rights reserved. TransNexus, NexSRS, NexOSS, Nex1, NextraNet, LookAhead Token, ClearIP, PeeringHouse, OSPrey and NexTransit are trademarks of TransNexus, Inc.

Table of Contents

1. INTRODUCTION 1
2. INSTALLATION AND CONFIGURATION 4
   - Platform Requirements 4
   - Installation Instructions....5
   - NexSRS Server Configuration....7
   - Upgrade Procedures 12
3. LICENSE KEY & FEATURES 15
   - Licensed Features....15
   - License Upgrade 16
   - NexSRS Server Features....18
4. CUSTOMERS, SUBNETS & VOIP DEVICES 20
   - Provisioning Customers 20
   - Adding Subnets....21
   - Adding VoIP Devices 23
5. DESTINATIONS, PRODUCTS & DIAL PLANS 28
   - Introduction 28
   - Define Destinations 29
   - Create Products 31
   - Dial Plans....38
   - Suggested Routes....43
   - ExtraNet Routing....43
6. NUMBER TRANSLATION 47
   - Pre-Routing Rules....49
   - Routing Called Number Manipulation Rules 53
   - Post-Routing Rules 53
7. NEXSRS SERVER OPERATION 54
   - Operating Scripts 54
   - Log Messages....55
   - Backing-up Configuration Files 56
   - Utilities 58
   - Watch Dog 61
8. CALL DETAIL RECORDS 62
   - Introduction 62
   - Current CDRs....66
   - NexSRS CDR Fields 67
   - Past CDRs 70

Managing Archived CDR Files 71
9. USER ADMINISTRATION 72
10. TROUBLESHOOTING 78
Routing Configuration 80
Features Configuration 85
Device Enrollment 86
CDR Management 88
License 89
400 Error Code – Bad Request 92
401 Error Code – Not Authorized 93
404 Error Codes – Route Not Found 95
503 Error Codes – Service Not Available 96
OSP Client Errors 97
ExtraNet 99
Questions and Answers 102
11. CISCO NOTES 107
12. APPENDICES 112
1. Release Notes 112
2. Load Balancing Algorithm 119
3. Dial Plan Selection Logic 121
4. NexSRS Routing Table – nexus.cfg 122
5. Dial Plan Assignment Configuration Files 125
6. Number Translation Configuration Files 126
7. Called Number Digit Filtering Configuration File 128
8. Termination Cause Codes 129
9. inter_osp_config.sh 132
10. Utility Scripts 133
11. SNMP_config.sh 137
13. INDEX 140

Chapter 1

Introduction

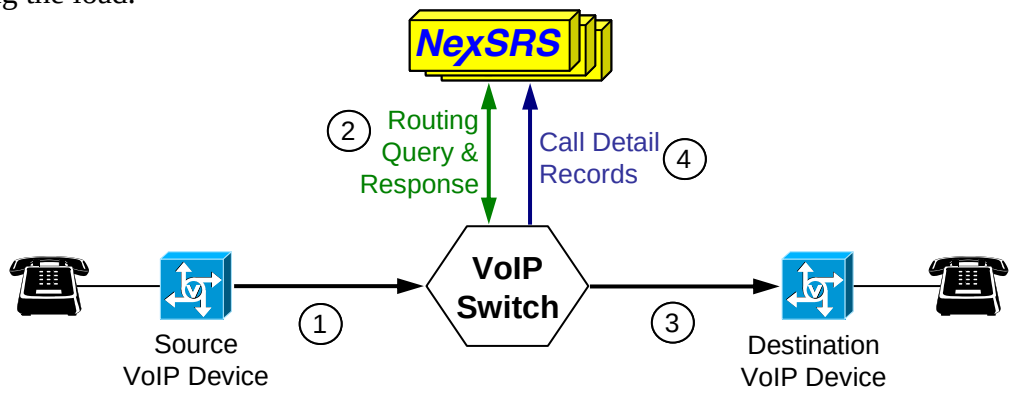
NexSRS Server
The NexSRS Server interfaces directly with VoIP signaling devices to provide real time routing information and Call Detail Record (CDR) collection. The NexSRS server uses either the ETSI OSP peering protocol or Cisco's GKTMP protocol to communicate with VoIP signaling devices (i.e. softswitch, SIP proxy, gatekeeper, session border controller). The NexSRS server is a stateless route server and may be distributed throughout the IP network to ensure high availability. The standard NexSRS server is available for Linux operating systems and a secure version of the NexSRS server which includes a certificate authority and supports SSL/TLS (Secure Sockets Layer/Transport Layer Security) is available for the SUN Solaris operating system (for sparc CPUs).

NexSRS features include:

- Routing based on source customer, source IP address, source trunk group, called number prefix, called number, calling number, time of day or day of week
- Rank order routing, percentage based load sharing, combination of rank order routing and load sharing
- Call blocking by customer, IP device, called number or calling number
- Partial call blocking from 0% to 100% by destination device
- Number translation based on source customer, source IP address, source trunk group, destination IP address, destination trunk group, calling number, called number, time of day or day of week
- Source IP address authentication
- ENUM client (additional module required)
- Clandestine CALEA routing and call auditing (additional module required)
- Call Detail Record collection
- Certificate server, digitally signed peering tokens (Solaris version only)
- Scalability - manage up 2,000,000 routes

NexSRS as a Route Server

NexSRS may be configured as a route server and CDR collection server for a VoIP signaling device such as a softswitch, gatekeeper, SIP proxy or session border controller. In this type of deployment, the NexSRS server provides the intelligence and scale for routing features that cannot be provided by a VoIP signaling device. For example, since many VoIP signaling devices cannot support a large least cost routing table, this function is off-loaded to the NexSRS server which can support a routing table with 2,000,000+ destinations. An example call scenario for this type of deployment is described below. A single VoIP Switch is shown in the diagram, but typically multiple VoIP signaling platforms of different types could are served in parallel by one or more NexSRS servers sharing the load.

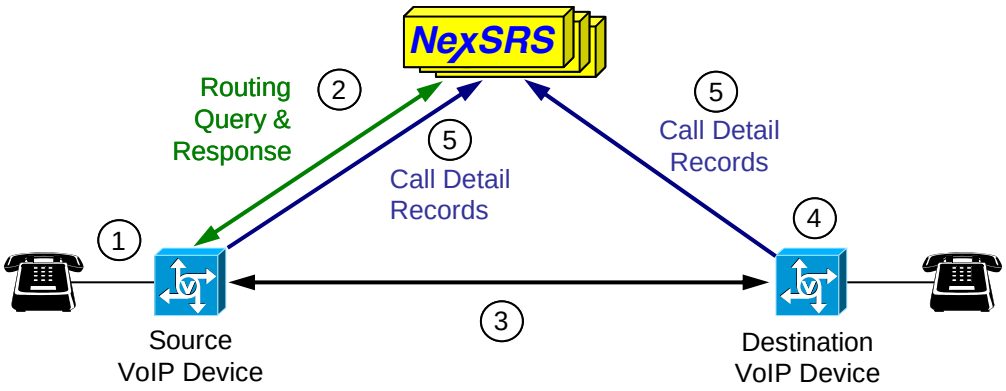


The following is a description of the call scenario in the previous diagram.

1. A call originates from the Source VoIP Device and is routed to the VoIP Switch.
2. The VoIP Switch sends a routing query to the NexSRS server. The NexSRS performs a route lookup and returns a list of IP addresses of devices which can complete the call to the called number. The NexSRS can translate the calling or called number as needed.
3. The call is routed from the VoIP Switch to the Destination VoIP Device which completes the call.
4. At the end of the call, the VoIP Switch sends a call detail record to the NexSRS server.

NexSRS as a Secure Peering Server

The NexSRS server may also be used as a secure peering server for authorizing and accounting for direct peer to peer communications between anonymous VoIP networks. For direct peering, each VoIP peer must enroll with the NexSRS certificate authority to obtain the NexSRS server's public key. Also, the NexSRS server signs the certificate request from each VoIP device. This feature requires the certificate authority and security features available only on the Solaris version of the NexSRS server. The call scenario in the diagram below illustrates the NexSRS server being used as a peering server.



1. A VoIP call originates from the Source VoIP Device.
2. The Source VoIP Device sends a route query to the NexSRS server which returns a list of IP addresses of devices which can terminate the call. Included with each address is a digitally signed peering token authorizing the call.
3. The Source VoIP Device sends a SIP Invite or Q.931 call setup message directly to the destination peer. The peering token is included in the SIP Invite or Q.931 call setup message.
4. The Destination VoIP Device validates the peering token with the NexSRS server's public key. If the token is valid, the Destination VoIP Device accepts the call.
5. When the call is finished, both the source and destination devices send call detail records to the NexSRS server.

Chapter 2

Installation and Configuration

The NexSRS Peering Server downloaded from the TransNexus web site is fully functional with an embedded demonstration license key for 600 calls per hour. Please see Chapter 3 for information on upgrading the NexSRS server license key for more than 600 calls per hour.

Platform Requirements

TransNexus recommends that the NexSRS server be run on Red Hat Enterprise Server or CentOS Linux. However, the NexSRS server has also been run other versions of Linux. Minimum hardware requirements are 1 GHz Intel CPU, 512 MB of RAM and a 4 GB hard drive. The Linux installation type must "server", not "workstation" or "personal desktop". For planning purposes, the NexSRS server can process approximately 60 calls per second per 1 GHz of CPU capacity. When the NexSRS server is used for routing and CDR collection with a Cisco gatekeeper, the expected performance is approximately 30 calls per second per 1 GHz of CPU capacity. For example, a server with dual 1.5 GHz CPUs would be able to process 180 calls per second. The equation for this example is:

{[(60 calls/second)/1 CPU GHz] * 2 CPUs * 1.5 GHz} = 180 calls/second

Memory requirements are driven primarily by the size of the routing table. For planning purposes, every 500,000 routes in NexSRS will require 1 GB of RAM. Service providers implementing domestic US inter-state and intra-state least cost routing should plan on needing 2GB or RAM for each NexSRS server.

User Interface

Viewing the NexSRS web interface requires Internet Explorer web browser 5.0 (or later) or FoxFire web browser 1.5 (or later). The NexSRS web interface also requires Java Runtime Environment 5.0 (or later) installed on the computer of the web browser. If Java is not installed, your web browser will automatically download and install the required version of Java.

Hardware Acceleration

The NexSRS Server can accommodate optional cryptographic acceleration hardware to speed-up cryptographic operations such as digitally signing OSP tokens. TransNexus recommends SafeNet CryptoSwift™ Secure Server Accelerator PCI cards (www.safenet-inc.com).

Hardware Configuration

The base time for the NexSRS server must be set to be the same base time as all VoIP devices that will use the NexSRS server since OSP peering authorization tokens have a life of +/- five minutes from the time when they are created. TransNexus recommends the NexSRS server base time be set to Universal Time Coordinates (UTC) using a Network Time Protocol (NTP) server common to all VoIP devices. Other applications may be run on the on the same Solaris platform with the NexSRS server, as long as those applications do not use ports 1080, 1081, 1082 and 1443.

There are no firm requirements for how the hard drive of a NexSRS Server should be partitioned. TransNexus provides the following partition examples as guidelines for new NexSRS server installations. The instructions throughout this User Manual assume that the NexSRS server is installed in a partition defined as /OSP.

Partitions	**Notes**	**36GB Hard Drive**	**72GB Hard Drive**
/	Root	5 GB	7 GB
Swap	Virtual Memory	Twice the amount of RAM	Twice the amount of RAM
/usr	Solaris utilities	5 GB	7 GB
/opt	Solaris utilities	5 GB	7 GB
/export/home	User directories	5 GB	7 GB
/OSP	NexSRS Server	14 GB	42 GB

Installation Instructions

This User Manual assumes that the directory where the NexSRS server is installed is softlinked to a link named ospnexus in the OSP partition. By updating the "ospnexus" softlink, the NexSRS administrator will be able to upgrade the NexSRS server without having to edit the .profile. However, this configuration is not a requirement. The user may install NexSRS server in any location.

Preparation Steps

1. Log-in to the host server with super-user access. This is required for adding a new user.
2. Create a group named "osp" and a user named "ospadmin". This group and user are used by scripts that simplify operation of the NexSRS server.
3 Create OSP partition.
4. Change ownership of OSP partition to ospadmin.

Installation Steps

1. Login into the host server prepared for NexSRS as user ospadmin. Navigate to the OSP partition.
2. Transfer the NexSRS software into the OSP partition.
3. Unzip the NexSRS.gz file.

4. Untar the NexSRS tar file. Untarring the file will create a new directory with the name `ospnexus_V.v.x` where `V.v.x` is the version of the NexSRS server installed.
5. Create a softlink named ospnexus that points to the new `ospnexus_V.v.x` directory.
6. Edit the .profile for the user "ospadmin" to include the variable $NexSRS_HOME which points to the current directory of the NexSRS server. The entry in the .profile based on the TransNexus recommendation is:

```
NexSRS_HOME=/OSP/ospnexus
export NexSRS_HOME
```

7. Move to the new directory by typing the command: `cd ospnexus.` Next, type: `ls` to get a listing of the new NexSRS server directory. The directory listing should be similar to what is shown below:

```
$ cd ospnexus
$ ls
ospnexus_2.8.0>ls
BACKUP                                          nexus.lic
CERTSTORE                                       nexus_log.stderr
LICENSE.txt                                     nexus_log.stderr.new
README.txt                                      nexus_log.stdout
RELNOTES.txt                                    nexus_server
auto_start_stop_osp_server.sh                   nexus_watch_dog
collect_debug_info.sh                           nexus.cdr
enum_config.sh                                  snmp_config.sh
etc                                             start_osp_server.sh
etc.loading                                     stop_osp_server.sh
etc.tree                                        uncompress_content.sh
exp.sh                                          uploaded_config.tmp
inter_osp_config.sh                             utils
libxerces-c1_7_0.so                             xitami
ospnexuscarrierservercommerciallicense.txt
```

The NexSRS server installation is now complete. The following is a brief description of some of the files and directories of the NexSRS server.

`nexus.cfg is` the NexSRS server configuration file with all routing information.

`nexus_watch_dog` program monitors the NexSRS server and Xitami web server and restarts these applications if needed. Otherwise the `nexus_watch_dog` program will automatically restart the NexSRS server.

`start_osp_server.sh` is the start script for the NexSRS server. This file also contains critical initialization parameters. The NexSRS server should only be started using this script.

`stop_osp_server.sh` is the stop script for the NexSRS server. The NexSRS server should only be stopped using this script. If the NexSRS server process is killed directly it will be automatically restarted by `nexus_watch_dog`.

`utils` is a directory which contains scripts which automate routine NexSRS maintenance operations such as compressing CDR files, deleting old CDR files, deleting old nexus.cfg backup files and cutting nexus.log files.

`xitami` is the directory for the Xitami web server that is included with the NexSRS server. This directory holds three important sub-directories.

`xitami/osp_ui/cdrs/` contains current and past CDR files.

`xitami/osp_ui/cdrs/ARCHIVE` contains archived CDR files.

`xitami/osp_ui/log` contains current and past log files for the NexSRS server.

Public Key Infrastructure Services

The NexSRS server is configured automatically at installation to support the following three SSL cipher suites:

SSL_RSA_WITH_RC4_128_SHA (0x00, 0x05)

SSL_DH_anon_WITH_RC4_128_MD5 (0x00 0x18)

SSL_RSA_WITH_DES_CBC_SHA (0x00, 0x09).

These cipher suites provide SSL interoperability with all OSP enabled VoIP devices. The particular cipher suite used for any individual SSL session is the result of negotiations between the SSL client and server. Other cipher suites are supported by the NexSRS server. Contact support@transnexus.com if you require custom configuration for other cipher suites.

If SSL is enabled on your NexSRS server, source and destination VoIP devices must use the following address format for the NexSRS server:

`https://<NexSRS server name or IP address>:1443/osp`

If SSL is not enabled on your NexSRS server, source and destination VoIP devices must use the following format for your settlement URL:

`http://<`NexSRS server name or IP address>:1080/osp

Certificate Authority Certificate

A certificate authority (CA) certificate is automatically generated when the NexSRS server is first installed. Its expiration date is ten years from the date of installation. To view the CA certificate of your NexSRS Server go to the following URL from Microsoft Internet Explorer: http://<hostname>:1080/scep/pkiclient.exe?operation=GetCACert where <hostname> is either the IP address or Domain name of the NexSRS server. A dialog box should come up asking if you want to open or save a file. Save the file to your PC with a .crt file name extension. The file will appear on your Windows PC as a certificate icon as shown to the right. Open this file with Internet Explorer to view the CA certificate details

pkiclient.crt

NexSRS Server Configuration

The NexSRS server requires minimal configuration. The following steps will prepare the NexSRS server for operation.

Avoid Conflicting Ports

Prior to starting the NexSRS server, ensure ports 1080, 1081, 1082 and 1443 are free. One technique to determine if these ports are free is to run the command: `telnet localhost 1080`. The following message should be displayed: `Unable to connect to remote host; connection refused`. Repeat this step for ports 1081 and 1443. If the message `Unable to connect to remote host; connection refused` is not displayed for each port, shut down the application(s) using these ports before starting the NexSRS server.

Initialize NexSRS Server Parameters

The `start_osp_server.sh` script contains important initialization parameters for the NexSRS server. These parameters may be changed by editing the `start_osp_server.sh` file using any text editor. The NexSRS server must be restarted for changes to this file to take affect.

The default values in this text file should not require changing except for the `ENGINE_ID_NUM` value. Every NexSRS server must be configured to have a unique engine ID number (0 -127) to ensure unique a Transaction ID for all OSP *AuthorizationResponses*. Each NexSRS server must be configured with a unique `ENGINE_ID_NUM`.

The `start_osp_server.sh` file has two sections. The first section contains commonly changed parameters that operators may want to change. The second section includes other parameters which should not be changed without consulting support@transnexus.com. The first section of the `start_osp_server.sh` file, with commonly changed parameters, is shown below.

```
#############################################################################
#
#                          start_osp_server.sh
#
#############################################################################
#
# This shell script is used to initialize parameters and start the
# NexSRS server.  The NexSRS server should always be started using this
# shell script.  Starting the NexSRS server application directly may
# result in errors due to incorrectly initialized variables.
#
# This file initializes the configuration parameters listed in the
# sections below.  For most cases, the only parameters in the COMMONLY
# CHANGED PARAMETERS section should be changed.  The default settings
# of the OTHER PARAMETERS have been configured for optimal performance
# and should only be changed after consulting support@transnexus.com.
#
# Please follow the procedures below to start the NexSRS server with
# this script.
#   a) Configure the parameters in this script and save the updated
#      file to the ospnexus directory
#   b) Ensure ports 1080, 1081, 1082 and 1443 are free.  If these ports
#      are being used by another application, there will be a conflict
#      with the NexSRS server.
#   c) Be logged in as ospadmin user
#   d) Be in the ospnexus directory
```

```
#   e) Type: ./start_osp_server.sh
#
#########################################################################
#
#                     COMMONLY CHANGED PARAMETERS
#
#########################################################################
#
# ENGINE_ID_NUM
#
# Each instance of the NexSRS Server should have a unique
# ENGINE_ID_NUM.  This parameter is used to ensure that the
# Transaction ID for each call authorized by the NexSRS server is
# unique in a network of NexSRS servers.  This parameter may be
# set to any integer from 0 to 127 inclusively.
#
# Set the desired engine ID number
#
ENGINE_ID_NUM=
export ENGINE_ID_NUM


#
# The AUTHENTICATE_ACTUAL_SOURCE_DEVICE flag controls the NexSRS
# server authentication policy for OSP AuthorizationRequests received
# from proxy devices.  If the flag is set to "Yes", the NexSRS server
# will authenticate both the proxy device and the actual source device
# which originated the call.  (In an OSP AuthorizationRequest from a
# proxy device, the Source device is reported in the DeviceInfo element
# of type transport or SourceAlternate element of type=h323).  This
# functionality can add additional security and/or offload
# authentication logic from proxy devices.  The default value is No.
#
AUTHENTICATE_ACTUAL_SOURCE_DEVICE=No


#
# The CDR_FILE_TIME_SPAN parameter determines how often the NexSRS
# server will create CDR files.  This parameter is defined in minutes.
# For example, if CDR_FILE_TIME_SPAN=60, the NexSRS server will write a
# new CDR file every 60 minutes.  Acceptable values for this parameter
# are 1, 2, 3, 4, 5, 6, 10, 12, 15, 20, 30 and 60.  If an unacceptable
# value is entered, the NexSRS server will use the default value of 60
# minutes.
#
CDR_FILE_TIME_SPAN=5


#
# CHECK_FOR_DUPLICATE_DESTINATIONS
#
# The server supports a global setting for specifying the policy for
# handling duplicate terminating devices.
#
```

```
# There are three possible settings:
#
# 1 - Never (Default).  The server will allow duplicate terminating
# devices to be returned in an authorization response.  This setting is
# preferred if calls are routed to a proxy device that does not support
# LookAhead routing.
#
# 2 - Always.  The server will always check for and remove all
# duplicate terminating devices.  This setting is preferred if calls
# are routed from a proxy device to a destination device.
#
# 3 - LookAhead Only.  The server will check and remove duplicates for
# destinations routed through look ahead enabled proxies only.  This
# setting is preferred for networks that have a mix of proxies -  some
# proxies support LookAhead routing and some proxies do not.
#
CHECK_FOR_DUPLICATE_DESTINATIONS=1
export CHECK_FOR_DUPLICATE_DESTINATIONS


#
# ENABLE_CRYPTO_HW_ACCELERATION
#
# The default installation of the NexSRS server assumes that
# cryptographic acceleration hardware is not being used.  If
# cryptographic acceleration hardware is being used, change this
# value to yes.
#
# NOTE: If this value is set 'yes' and the server is not equipped
# with crypto hardware, or if the crypto hardware is not properly
# configured, the NexSRS application may not operate properly.
#
ENABLE_CRYPTO_HW_ACCELERATION=no
export ENABLE_CRYPTO_HW_ACCELERATION


# MAXIMUM_CALL_LENGTH
#
# This parameter is a global variable that defines the maximum call
# duration authorized by the NexSRS server.  This value is embedded in
# the OSP token and transmitted in the call set-up to the terminating
# network.  When the duration of a call exceeds the MAXIMUM_CALL_LENGTH
# the terminating network should terminate the call.  The default value
# of MAXIMUM_CALL_LENGTH is 14400 seconds (4 hours).  The allowable
# range of values for MAXIMUM_CALL_LENGTH is 0 to 7776000 seconds
# (90 days: 60*60*24*90=7,776,000).
#
MAXIMUM_CALL_LENGTH=14400
export MAXIMUM_CALL_LENGTH


# AVAILABLE_DISK_SPACE_WARNING_THRESHOLD
#
# The NexSRS server monitors the amount of disk space available in the
```

```
# disk partition where the NexSRS server is installed.  This parameter
# is measured in megabytes of disk space and the default value is
# 1,000 MB.  The minimum value for this parameter is 1 MB.  When
# available disk space falls below the threshold set by this parameter
# a warning message will be written to the NexSRS server log file.
#
AVAILABLE_DISK_SPACE_WARNING_THRESHOLD=1000
export AVAILABLE_DISK_SPACE_WARNING_THRESHOLD


# AVAILABLE_DISK_SPACE_CHECK_TIME_INTERVAL
#
# This parameter defines how often the NexSRS server checks available
# disk space. The default value is ten minutes and the minimum is one
# minute.
#
AVAILABLE_DISK_SPACE_CHECK_TIME_INTERVAL=10
export AVAILABLE_DISK_SPACE_CHECK_TIME_INTERVAL


# AVAILABLE_DISK_SPACE_REQUIRED_THRESHOLD
#
# When available disk space falls below the threshold set by this
# parameter the NexSRS server will shutdown.  This parameter is
# measured in megabytes and the default value is 15 MB.  The minimum
# value is 0 MB.  This parameter must be set to be less than or equal
# to the AVAILABLE_DISK_SPACE_WARNING_THRESHOLD.
#
# The recommended value for this parameter is 2000 bytes * maximum
# calls per minute * AVAILABLE_DISK_SPACE_CHECK_TIME_INTERVAL.
# For example, if the maximum number of calls per minute is 1000
# and AVAILABLE_DISK_SPACE_CHECK_TIME_INTERVAL = 10 minutes,
# then the recommended threshold is:
#   2000 bytes * 1000 calls per minute * 10 minutes = 20 MB
#
AVAILABLE_DISK_SPACE_REQUIRED_THRESHOLD=15
export AVAILABLE_DISK_SPACE_REQUIRED_THRESHOLD
```

Starting the NexSRS Server

Once the installation and configuration steps above are completed, the NexSRS server is ready to start. **The NexSRS server must always be started using the `start_osp_server.sh` script**. This shell script is required to initialize parameters configured for the NexSRS server. The operator must own the nexus_server application and be located in the `$NexSRS_HOME` directory in order to start the NexSRS server. Type this command to start the NexSRS server: `./start_osp_server.sh`

To check if the server has started, type this command: `ps -ef|grep nexus`

If the NexSRS server is running you will see both the `nexus_server` and `nexus_watch_dog` applications are alive. Below is an example of what to expect if the NexSRS server is operating:

```
>ps -ef|grep nexus
```

```
root 26612     1  0   Nov 19 ?         0:00 ./nexus_watch_dog
root 26617 26612  0   Nov 19 ?       652:23 nexus_server
```

Stopping the NexSRS Server

Use the `stop_osp_server.sh` script to stop the NexSRS server. This script automates the process of shutting down the NexSRS server and making certain the NexSRS server process has exited. The user of this script must own the nexus_server application.

To run the script type `./stop_osp_server.sh` from the $NexSRS_HOME directory.

The script will ask the NexSRS server to stop and print `NexSRS server has stopped` when the NexSRS server is shutdown. If the NexSRS server is not running when the script is run, the user will see the message `NexSRS server is not running`.

Upgrade Procedures

To simplify the upgrading of NexSRS servers which are in production operation, TransNexus recommends that the following procedures be followed to minimize the risk of service interruption or lost call detail records. These procedures apply for upgrading two NexSRS servers (named NexSRS1 and NexSRS2) that are operating in parallel. The procedures also assume that NexOSS config_push and cdr_pull applications are in operation to push new configurations and collect CDRs. However, these generic instructions can apply to any number of NexSRS servers. Following these instructions will ensure there is no interruption of service and no CDRs lost during the upgrade process.

Preparation

1. Copy the tar file with the new version of NexSRS server to the NexSRS1 partition or directory which contains the ospnexus soft link (one level above $NexSRS_HOME). This is the /OSP partition in the TransNexus recommended configuration. Perform a UNIX list command to verify the tar file was successfully copied as shown in the example below.
2. Note the ENGINE IDs for both NexSRS1 and NexSRS2. Each NexSRS server must have a unique ENGINE ID to avoid duplicate Transaction IDs. These ENGINE IDs are found in the `start_osp_server.sh` script and must be re-configured before starting the upgraded NexSRS servers.
3. Obtain a new software license from TransNexus. Check the `"NotAfterVersion"` field of the NexSRS server software license to make certain the new version is licensed for use. For example, if the `"NotAfterVersion"` is 1.5.0, then NexSRS server version 1.5.1 will not operate when installed because it is not licensed.
4. On the NexSRS web interface, go to the **Backup** page under the **Server** tab. Use the Back-up links to back-up a copy of each of the ten current NexSRS configuration files to a local PC. Note the location of the back-up nexus.cfg. It will be used later to update the configuration of the new installation.

5. If the NexOSS config_push and cdr_pull applications are scheduled to run automatically, stop these processes. Running these processes during the installation procedure might cause errors.

Stop NexSRS1.

6. On NexSRS1, go to the Features page, under the **Server** tab, of the user interface and turn off routing. VoIP devices will detect routing has been turned off on NexSRS1 and will begin shifting OSP *AuhorizationRequests* to NexSRS2. Wait several minutes and check the log messages of both servers. The call attempts per minute on NexSRS1 should fall to 0. The call attempts per minute on NexSRS2 should increase since VoIP devices which were using NexSRS1 will be sending *AuthorizationRequests* to NexSRS2.
7. On NexSRS1, go to the Features page, under the **Server** tab, of the user interface and turn off CDR Collection. Wait until the next cdr file is written and manually run `cdr_pull` on the NexOSS platform. (The time when a new cdr file is written is determined by the parameter CDR_FILE_TIME_SPAN.) This step will ensure that all cdr records are removed from NexSRS1 server before it is upgraded. Go to the **Call Details** tab on the user interface and make certain that all past cdr files have been collected from NexSRS server 1 and that the current cdr file has 0 records.
8. Stop NexSRS1. Go to the $NexSRS_HOME directory server and run the stop_osp_server.sh script to stop the NexSRS server.

Install and Configure the New NexSRS server

9. Make certain group “osp” and user “ospadmin” are defined and being used to run and maintain the NexSRS1. Log in as ospadmin and make sure that the user has the right to create, read, write and execute files.
10. Untar and unzip the new NexSRS server on NexSRS1.
11. Change ospnexus softlink to point to new NexSRS server directory.
12. Copy xitami configuration files xitami.aut and iplist.txt to the new NexSRS server directory.
13. Edit the `start_osp_server.sh` script to make certain all configuration variables are defined correctly.

Start the New NexSRS Server

14. Start the new NexSRS server 1 installation by running the `start_osp_server.sh` script.
15. Update the NexSRS software license using the upgrade link on the License page of the NexSRS user interface.
16. Restore the nexus.cfg configuration, backed-up in step 4, using the user interface of the NexSRS server.

17. Monitor both NexSRS1 and NexSRS2. Traffic will return to NexSRS1 as OSP clients open new connections to NexSRS1, and as existing connections to NexSRS2 fail-over to NexSRS1.
    a. Monitor the log messages of both NexSRS servers by viewing the Logs page, under the **Server** tab, on the user interface. Alternatively, log messages may be viewed by using the following UNIX command from the $NexSRS_HOME directory:
       `$ tail -50f xitami/osp_ui/log/nexus.log`
    b. Monitor the CDRs of both NexSRS servers by viewing the Current page, under the **Call Details** tab of the user interface.

18. Once the traffic has moved back to NexSRS1, perform the following steps:
    a. If step 17 verifies the upgrade is successful, re-configure the NexOSS applications config_push and cdr_pull to run automatically. (These processes were shutdown in step 5.)
    b. If step 17 indicates the upgrade was not successful, proceed to the Roll-Back instructions below to restore the original version of the NexSRS server.

To upgrade NexSRS2, repeat steps 1 – 18 for NexSRS2.

How to Roll-Back

If for some reason the upgrade procedure is unsuccessful, or it is necessary to roll-back to the pre-upgrade version NexSRS server, follow these procedures.

1. Stop the NexSRS server – go to the current $NexSRS_HOME directory and run "`stop_osp_server.sh`"

2. Go to the directory where the previous version of NexSRS server is installed.

3. Change the softlink, changed in step 11 above, back to its original configuration.

4. Start the NexSRS server – go to the current $NexSRS_HOME directory and run `start_osp_server.sh`.

5. Re-configure the NexOSS applications config_push and cdr_pull to run automatically. (These processes were shutdown in step 5 of the upgrade procedure.)

Chapter 3

License Key & Features

The NexSRS Server download software includes an embedded evaluation license key that is valid for any IP address. The maximum performance with the evaluation license is to 600 calls per hour. A free registration license key is for 3000 calls per hour is available to each company which registers its NexSRS server. A commercial license key is required to increase the NexSRS Server call processing capacity above 3000 calls per hour.

Licensed Features

Below is a description of each licensed feature of the NexSRS server and a screen shot of the license page of a NexSRS server.

Version is the name, software version and build number of the NexSRS server installed.

Licensed to value specifies the host IP assigned to the installed license. Each commercial license provided by TransNexus is assigned to one or more host IP addresses.

Maximum Devices is the maximum number of VoIP devices that can be defined in the NexSRS server routing table.

Busy Hour Call Attempts is the maximum number of OSP Authorization Request messages that the NexSRS server will process in an hour. An OSP Authorization Request message is generated each time a Source VoIP Device queries the NexSRS server for routes and authorization to complete a VoIP call to a Destination VoIP Device.

Maximum Number of Digits is the maximum number of digits configurable for a route.

Pre-Routing Called Number Digit Filtering automatically blocks calls mis-dialed called numbers that are too long or too short.

Pre-Routing Called Number Translation allows use of the called number translation feature before searching for a route.

Routing Called Number Manipulation enables the called number to be modified for the routing algorithm, but restores the called number to its original value after the routing algorithm lookup is completed.

Post-Routing Called Number Translation allows use of the called number translation feature after searching for a route.

Post-Routing Calling Number Translation allows use of the calling number translation feature after searching for a route.

Number of Route Plan Assignment Sets defines the number of different route plans that may be used in sequential order to find a route for each call.

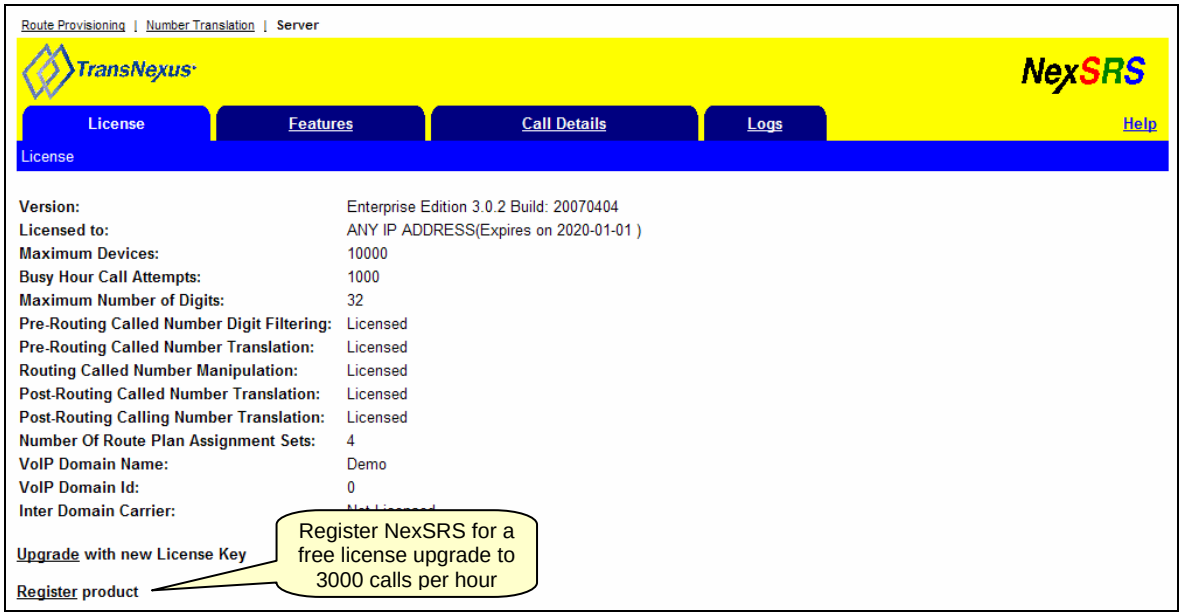
VoIP Domain Name is the name of the customer to whom the license has been issued.

VoIP Domain ID is a unique ID assigned to every licensed customer.

Inter Domain Carrier is the name, and NexSRS server IP addresses, of the inter-domain carrier serving the NexSRS server. If this license field is populated, the NexSRS server will automatically query the inter-domain carrier OSP servers for destination routes when necessary. See Chapter 4 for more information about ExtraNet routing to an OSP enabled inter-domain carrier.

Evaluation License Key

The licensed features of the NexSRS server can be viewed on the License page under the **Server** tab of the NexSRS server user interface. The following is a screen shot of the License page displaying the features licensed with the demonstration license key. The evaluation license key enables the NexSRS server to run on a server with any IP address and manage 600 calls per hour with full functionality until 2020-01-01.


Route Provisioning | Number Translation | Server
TransNexus
NexSRS
License
Features
Call Details
Logs
Help
License
Version: Enterprise Edition 3.0.2 Build: 20070404
Licensed to: ANY IP ADDRESS(Expires on 2020-01-01)
Maximum Devices: 10000
Busy Hour Call Attempts: 1000
Maximum Number of Digits: 32
Pre-Routing Called Number Digit Filtering: Licensed
Pre-Routing Called Number Translation: Licensed
Routing Called Number Manipulation: Licensed
Post-Routing Called Number Translation: Licensed
Post-Routing Calling Number Translation: Licensed
Number Of Route Plan Assignment Sets: 4
VoIP Domain Name: Demo
VoIP Domain Id: 0
Inter Domain Carrier:
Upgrade with new License Key
Register product
Register NexSRS for a free license upgrade to 3000 calls per hour

License Upgrade

To upgrade the call capacity of your NexSRS peering server to more than 3000 calls per hour requires a commercial license key. The commercial license key is a text file that you will receive via e-mail from TransNexus. The following is an example license key.

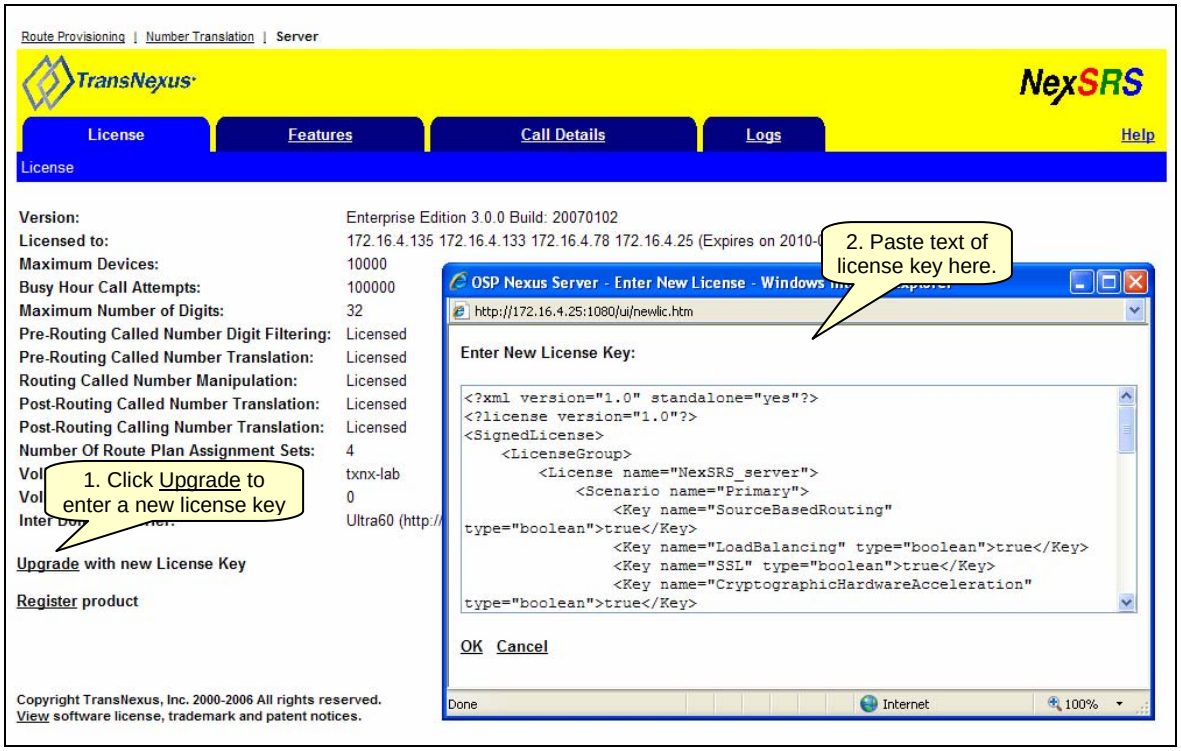
```
<?xml version="1.0" standalone="yes"?>
<?license version="1.0"?>
<SignedLicense>
    <LicenseGroup>
        <License name="NexSRS_server">
            <Scenario name="Primary">
                <Key name="SourceBasedRouting" type="boolean">true</Key>
                <Key name="LoadBalancing" type="boolean">true</Key>
                <Key name="SSL" type="boolean">true</Key>
                <Key name="CryptographicHardwareAcceleration" type="boolean">true</Key>
                <Key name="PreRoutingCalledNumberDigitFiltering"
type="boolean">true</Key>
                <Key name="PreRoutingCalledNumberTranslation" type="boolean">true</Key>
                <Key name="RoutingCalledNumberManipulation" type="boolean">true</Key>
                <Key name="PostRoutingCalledNumberTranslation"
type="boolean">true</Key>
                <Key name="PostRoutingCallingNumberTranslation"
type="boolean">true</Key>
                <Key name="BusyHourCallAttempts" type="number">360000</Key>
                <Key name="Devices"              type="number"> 10000</Key>
                <Key name="E164Resolution"       type="number">    32</Key>
                <Key name="VoIPDomainId"         type="number">     0</Key>
                <Key name="NumberOfRoutePlanAssignmentSets" type="number">4</Key>
                <Key name="VoIPDomainName"         type="string">Demonstration</Key>
                <Valid type="IP">129.186.11.164</Valid>
                <Valid type="IP">129.186.11.106</Valid>
                <Valid type="NotBeforeVersion">1.0.0</Valid>
                <Valid type="NotAfterVersion"> 3.0.0</Valid>
                <Valid type="NotBeforeDate">2000-01-01</Valid>
                <Valid type="NotAfterDate"> 2010-01-01</Valid>
            </Scenario>
        </License>
    </LicenseGroup>
    <Signature>
        MCwCFHPYprNvNE3uuwXNhDVKiF2A3mOiAhOkeFkvX+nWV6S1OL+O7VMCG0lkaq==
```

TransNexus recommends that a back-up copy of the commercial license.txt file be stored by the NexSRS server administrator.

The example commercial key above licenses the use of the following features: Source Based Routing, Load Balancing, SSL and Cryptographic Hardware Acceleration. If these features were not licensed, their license value would be false instead of true. Licensed operating capacity is set to 360,000 Busy Hour Call Attempts for 10,000 different Devices with E.164 Resolution routing capabilities up to 32 digit dial strings. This license enables two NexSRS servers running on IP addresses 129.186.11.164 and 129.186.11.106, is valid for any version of the NexSRS server from 1.0.0 to 2.11.0 and for use anytime between 1 January 2000 and 1 January 2010.

If you are running a free evaluation version of the NexSRS Peering Server, click on the **Register product** link on the license page to register your software with TransNexus to receive a free commercial license key to upgrade the capacity of your NexSRS Peering Server to 3000 call attempts per hour.

The commercial license key is installed using the **Upgrade with new License Key** link on the License page under the **Server** tab of the NexSRS server user interface. Clicking on the **Upgrade with New License Key** link will cause the **Enter New License Key** pop-up box to appear as shown below.

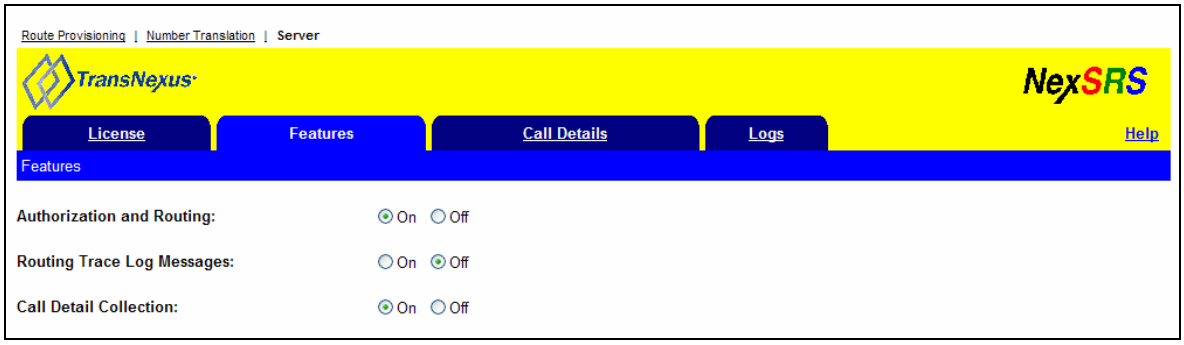


Highlight the text of the commercial license key from the .txt file you received and copy and paste the commercial license into the pop-up box on the NexSRS server user interface and then click OK. It is critical that the commercial license key be carefully cut and paste. Any missing characters or additional characters will cause the NexSRS server to reject the license key. The feature set and operating capacity of the NexSRS server are automatically upgraded and displayed on the License page when a new license key is installed.

If the feature set and operating capacity of the NexSRS server are not automatically upgraded then the NexSRS server has not accepted the license key. If this occurs, first view the log messages on the NexSRS server for any error messages. Double check that the license key was cut and pasted properly without any missing or additional text. Verify that the licensed IP addresses, NotBeforeVersion/NotAfterVersion and NotBeforeDate/NotAfterDate values are correctly set for your server. Please refer to the Appendix for examples of rejected log messages, the resulting log messages and suggested remedy. If these steps do not reveal the problem, paste the license key into an e-mail and contact support@transnexus.com. Please include a copy of any NexSRS server log messages in your e-mail to TransNexus support.

NexSRS Server Features

The three features: Authorization and Routing, Routing Trace Log Messages and Call Detail Collection may be turned On or Off using the radio buttons shown below on the **Features** page of the NexSRS server user interface.



Authorization and Routing is the function of responding to an OSP *AuthorizationRequest* from a VoIP device and providing an OSP *AuthorizationResponse*. When Authorization and Routing is enabled, the NexSRS server will accept an OSP *AuthorizationRequest* from a source VoIP device, authenticate that the source VoIP device is a valid device and then respond with an OSP *AuthorizationResponse* which includes a list of destination VoIP devices, and corresponding OSP authorization tokens, which can complete the VoIP call to the called number.

If Authorization and Routing is disabled, OSP *AuthorizationRequest* messages from source VoIP devices are rejected and the source VoIP device will send the OSP *AuthorizationRequest* to another NexSRS server. Authorization and Routing might be disabled, for example, if another platform manages these functions, and the NexSRS server is used only for collecting Call Detail Records or OSP *UsageIndication* messages.

Routing Trace Log Messages is a feature that provides detailed logging of each step within the routing algorithm. This feature is useful training tool to help to understand the details of complex routing plans. This feature is most useful when call volume is very low and the step by step routing details can be observed for individual calls. This feature is turned off by default and should not be activated in production operation since it creates a vast quantity of log messages which could quickly consume all available hard disk space.

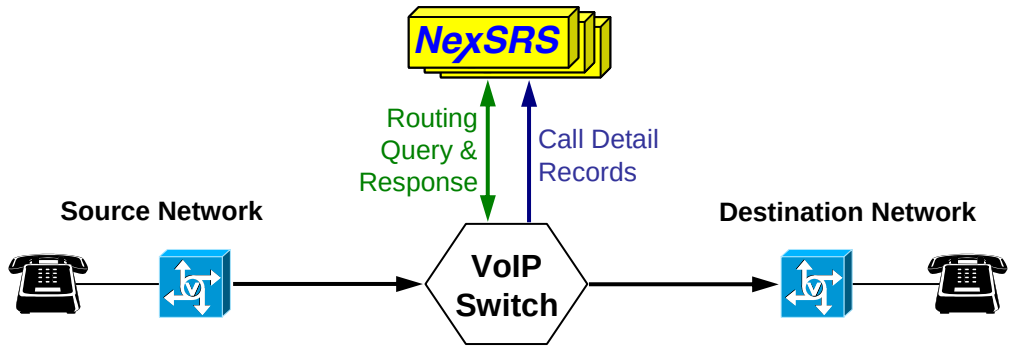
Call Detail Collection is the function of collecting Call Detail Records (CDRs) or OSP *UsageIndication* messages from both source and destination VoIP devices. If Call Detail Collection is disabled, CDRs, from the source and destination VoIP devices are not accepted by the NexSRS server. If a source or destination device sends a CDR to an NexSRS server and the CDR is not accepted, the source or destination device will attempt to deliver the CDR to another NexSRS server until the CDR is successfully confirmed as received by an NexSRS server.

Chapter 4

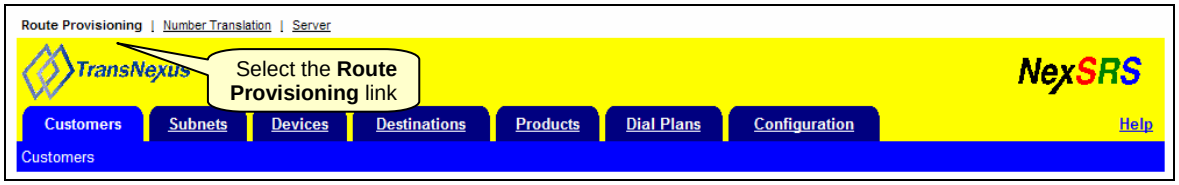
Customers, Subnets & VoIP Devices

Provisioning Customers

The first step in provisioning your NexSRS server is to provision Customers. In NexSRS, logical groups of VoIP devices are configured as Customers. Each Source VoIP network, (a network that originates calls) is configured as a Customer. Each Destination VoIP network, telecom vendor, or carrier that only terminates VoIP traffic is configured as a Customer. VoIP networks that both originate and terminate calls are configured as a Customer. Also, the operator of the NexSRS should also be configured as a Customer. In the diagram below, both the source and destination networks must be provisioned as Customers. In addition, the operator of the NexSRS server and VoIP Switch must also be configured as a Customer.

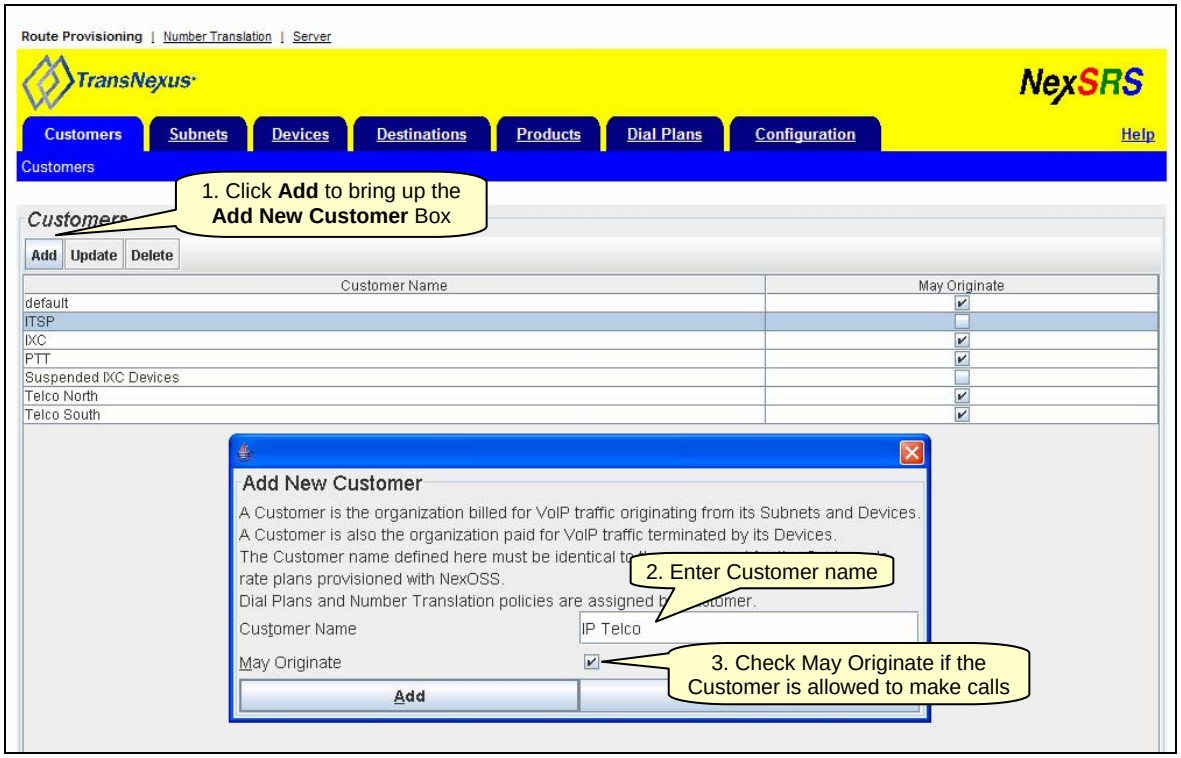


The first step in provisioning Customers is to select the Route Provisioning application of the NexSRS server at the top of the web interface as shown below.



The following screenshot is the Customers page on the Route Provisioning web interface. The first step to add a Customer is to click the **Add** button to make the **Add New Customer** box appear. A new Customer is added as shown in step 2. Customer names may be any combination of alphanumeric characters, the forward slash (/), underscore (_) and the dash (-). The Customer name may be up to 25 characters long and may include single spaces between

characters. Once a Customer has been added to the NexSRS server, the Customer may be renamed or deleted by using the **Update** or **Delete** buttons on the **Customers** page.

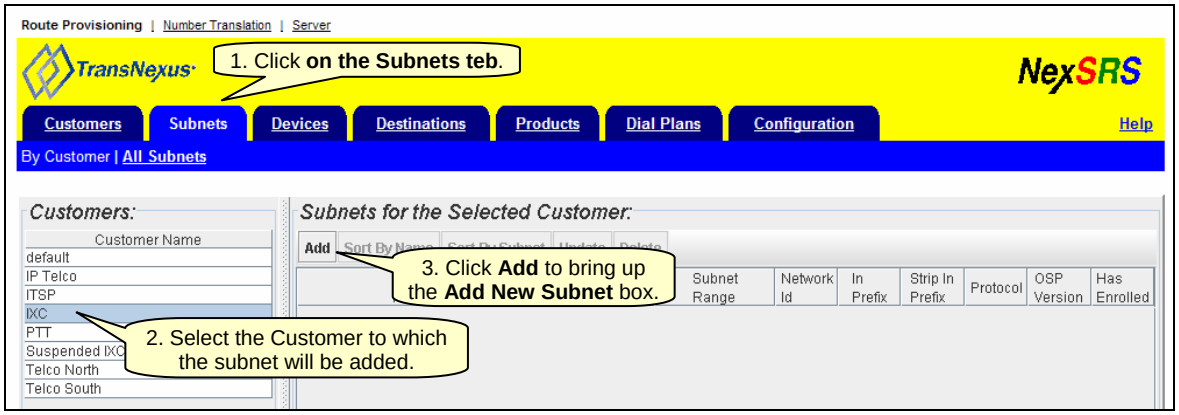


Another feature to note in the previous screenshot is the May originate checkbox. The status of this box indicates and controls whether or not a Customer may or may not originate VoIP calls. Unchecking the May originate status is a convenient way to temporarily turn-off a customer's rights to originate traffic without altering the routing table. For example, in the screenshot above, all Customers may originate traffic except the 'ITSP' and the 'Suspended IXC Devices' Customers.

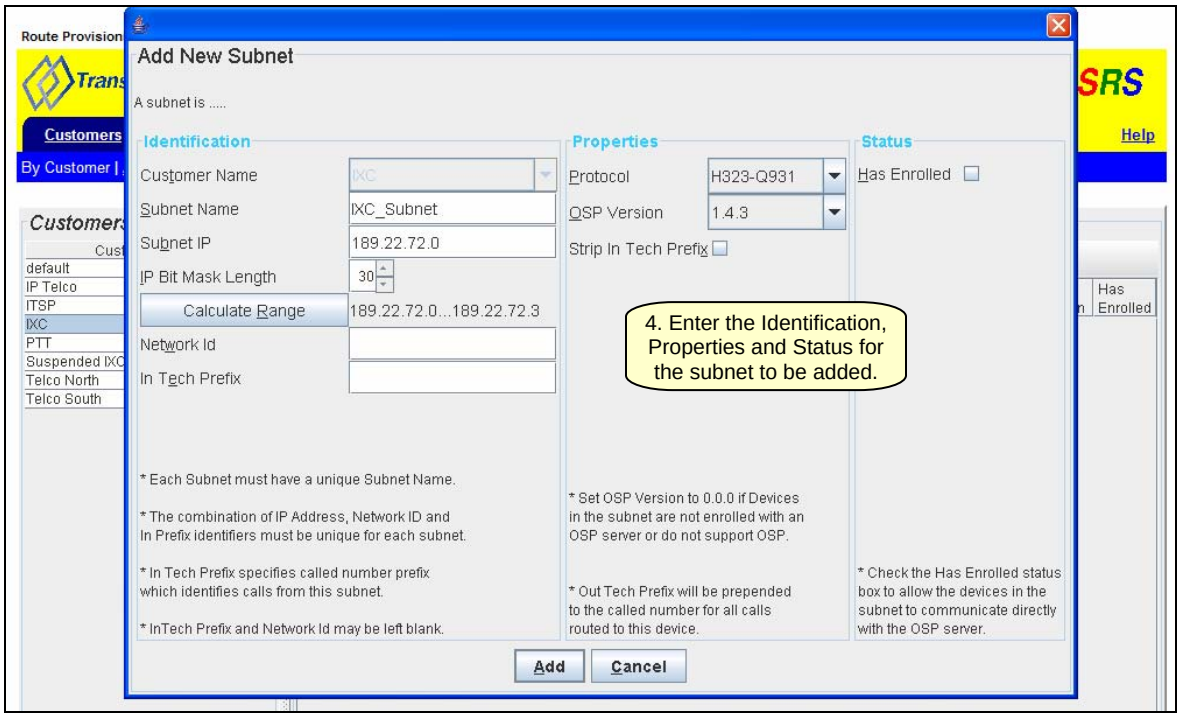
Adding Subnets

A subnet is a sub-network of continuous IP addresses within in a specific range. The number of IP addresses in a subnet can range from 2 to 2,147,483,648 IP addresses. Calls may be originated from any VoIP device with a subnet IP address. However, calls may not be routed to a subnet, since a VoIP call must be routed to a specific IP address.

The screenshot below illustrates how to add a VoIP subnet. First select the **Subnets** tab and then click on By Customer to view subnets by Customer or All to view all subnets. In the screenshot below, the IXC Customer has been selected (note the IXC Customer in the left column is highlighted) and the **Add** button has clicked to activate the **Add New Subnet** pop-up window.



The **Add New Subnet** box appears as shown below. The three type of information which may be entered for each subnet are Indentification, Properties and Status.


Add New Subnet
A subnet is .....
Identification
Customer Name
IXC
Subnet Name
IXC_Subnet
Subnet IP
189.22.72.0
IP Bit Mask Length
30
Calculate Range
189.22.72.0...189.22.72.3
Network Id
In Tech Prefix
Properties
Protocol
H323-Q931
OSP Version
1.4.3
Strip In Tech Prefix
Status
Has Enrolled
4. Enter the Identification, Properties and Status for the subnet to be added.
* Each Subnet must have a unique Subnet Name.
* The combination of IP Address, Network ID and In Prefix identifiers must be unique for each subnet.
* In Tech Prefix specifies called number prefix which identifies calls from this subnet.
* InTech Prefix and Network Id may be left blank.
* Set OSP Version to 0.0.0 if Devices in the subnet are not enrolled with an OSP server or do not support OSP.
* Out Tech Prefix will be prepended to the called number for all calls routed to this device.
* Check the Has Enrolled status box to allow the devices in the subnet to communicate directly with the OSP server.
Add
Cancel

Subnet Identification:

The combination of identification fields must be unique for each subnet.

Customer Name – The name of the Customer to which the subnet is assigned. This field is grayed-out in the screenshot above since the IXC Customer has already been selected.

Subnet Short Name – A descriptive alpha-numeric name for the subnet. The short name must be unique. Two subnets cannot share the same short name.

Subnet IP – Enter the first IP address in the range of subnet addresses.

IP Bit Mask Length – The subnet IP base address and subnet mask determines what range of the 32-bit Internet belongs to a given subnet. The IP Bit Mask Length defines how many

bits are in the bit mask. The IP Bit Mask Length may be as long as 31 (defining a range of two IP addresses) to as short as 1 (defining a range of 2^{31} IP addresses).

Calculate Range – After entering the IP Bit Mask Length, click on the Calculate Range button to determine the range of IP addresses in the subnet.

Network ID – This field is used to partition a subnet. The combination of subnet range and Network ID must be unique. Two different subnets may be defined with the same range of IP addresses if the two subnets have different Network IDs.

In Tech Prefix – This field defines an expected called number prefix and is used to partition a subnet. The combination of subnet range and In Tech prefix must be unique. Two different subnets may be defined with the same range of IP addresses if the two subnets have different In Tech prefixes.

Subnet Properties

Protocol – From the drop down menu, select the call signaling protocol used by subnet devices. The choices for are:

- h323-Q931. - for h.323 gateways
- h323-LRQ - for h.323 gatekeepers
- SIP - for SIP devices
- IAX – Inter-Asterisk eXchange protocol

OSP Version – Select the version of OSP supported by subnet devices.

- 0.0.0 – device does not support OSP
- 1.4.3 – appropriate for any OSP enabled device
- 2.1.1 – for devices which support version 2 OSP features
- 2.1.1-P – for proxy devices which support LookAhead routing

Strip In Tech Prefix – If this box is checked, the NexSRS server will always translate the called number to remove the In Tech prefix.

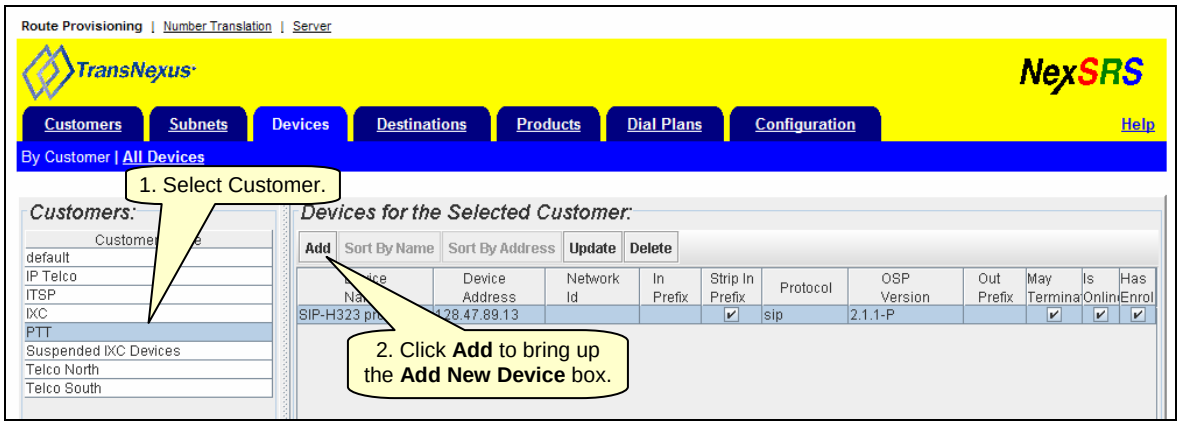
Subnet Status

Has Enrolled – If this box is unchecked, call routing requests from subnet devices will be denied unless the requests are sent via a proxy device (i.e. session border controller, IP to IP gateway, gatekeeper or SIP proxy) that has enrolled with the NexSRS server.

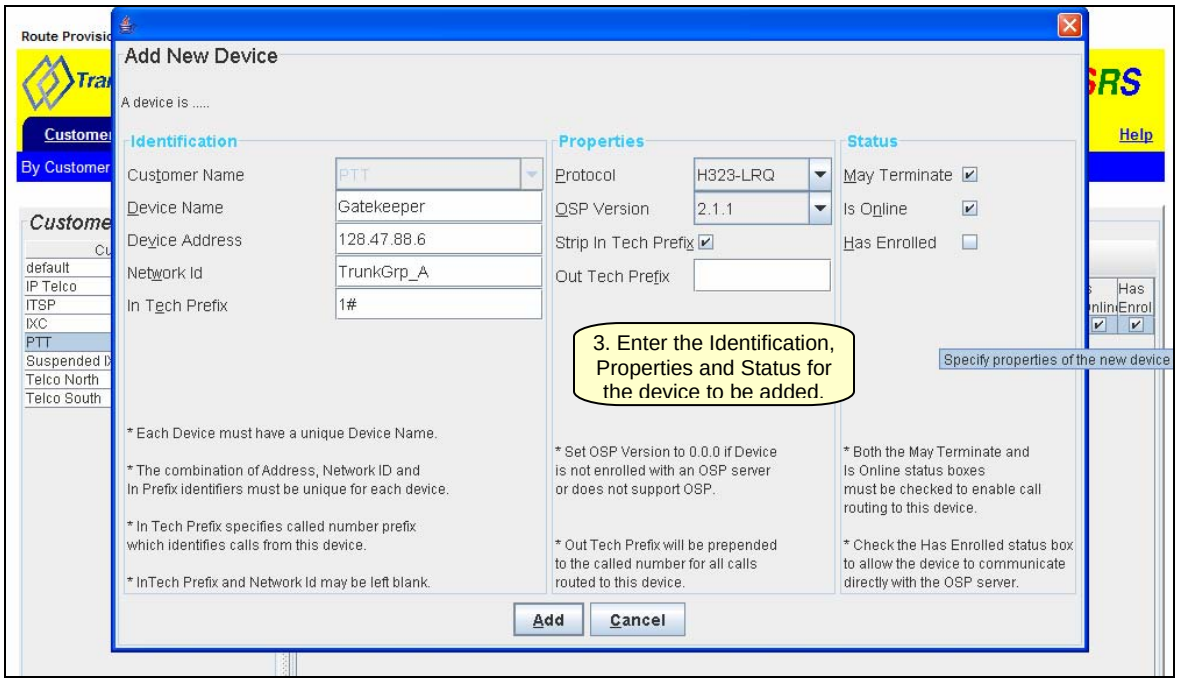
Adding VoIP Devices

In NexSRS a VoIP device may be any device which handles VoIP signaling, such as a VoIP to PSTN gateway, a softswitch, a session border controller, a SIP proxy, a gatekeeper or a softphone. Each VoIP device configured in NexSRS must be assigned to a Customer. A VoIP device cannot not be assigned to multiple Customers. There are no limits regarding the number of devices allowed per Customer. A Customer may have zero VoIP devices assigned to it, or a Customer may have the maximum number of devices allowed by the commercial license. The routing and number translation policies created for each Customer apply to all devices assigned to the Customer.

VoIP devices are added to the NexSRS server configuration through the **Devices** page of the NexSRS Server user interface. To add a device to the NexSRS server, first select the Customer for which you want to add a device. For example, in the following screenshot, the PTT Customer has been selected and is highlighted. Next, click on the **Add** button on the **Devices** page to activate the **Add New Device** box.



The **Add New Device** box appears as shown below. The three type of information which may be entered for each device are Indentification, Properties and Status.


Add New Device
A device is .....
Identification
Customer Name
PTT
Device Name
Gatekeeper
Device Address
128.47.88.6
Network Id
TrunkGrp_A
In Tech Prefix
1#
Properties
Protocol
H323-LRQ
OSP Version
2.1.1
Strip In Tech Prefix
Out Tech Prefix
Status
May Terminate
Is Online
Has Enrolled
3. Enter the Identification, Properties and Status for the device to be added.
Specify properties of the new device
* Each Device must have a unique Device Name.
* The combination of Address, Network ID and In Prefix identifiers must be unique for each device.
* In Tech Prefix specifies called number prefix which identifies calls from this device.
* InTech Prefix and Network Id may be left blank.
* Set OSP Version to 0.0.0 if Device is not enrolled with an OSP server or does not support OSP.
* Out Tech Prefix will be prepended to the called number for all calls routed to this device.
* Both the May Terminate and Is Online status boxes must be checked to enable call routing to this device.
* Check the Has Enrolled status box to allow the device to communicate directly with the OSP server.
Add
Cancel

Device Identification:
The combination of identification fields must be unique for each device.

Customer Name – The name of the Customer to which the device is assigned. This field is grayed-out in the screenshot above since the PTT Customer has already been selected.

Device Short Name – A descriptive alpha-numeric name for the device. The short name must be unique. Two devices cannot share the same short name.

Device IP – Enter the IP address of the VoIP device. If a domain name is entered as the Device Address, the NexSRS server will perform a DNS query to its domain name server to obtain the IP address for the domain name of the new device. If the domain name cannot be resolved by the DNS query, the device will not be added to the NexSRS server configuration and the following two log messages will be recorded.

```
2001-10-14 23:01:14|  1|Unable to add device
2001-10-14 23:01:14|  1|Failed to resolve IP address for <Newco.com>
```

If the IP address or domain name of the device being added has already been configured in the NexSRS server to another device, the new device will not be accepted and the NexSRS server will indicate that the IP Address or DNS name is already in use.

Network ID – This field is used to partition a device. The combination of device IP, Network ID and In Tech Prefix must be unique. A device may be configured more than once with the same IP address if the device indentification has different Network IDs or In Tech Prefixes. For example, a large VoIP gateway can be partitioned into several logical gateways, based on Network ID.

In Tech Prefix – This field defines an expected called number prefix and is used to partition a device. The combination of IP address, Network ID and In Tech prefix must be unique. A device may be configured with the same IP address more than once if each device configuration has a different In Tech Prefix.

Device Properties

Protocol – From the drop down menu, select the call signaling protocol used by the device. The choices for are:

- h323-Q931. - for h.323 gateways
- h323-LRQ - for h.323 gatekeepers
- SIP - for SIP devices
- IAX – Inter-Asterisk eXchange protocol

OSP Version – Select the version of OSP supported by the device.

- 0.0.0 – device does not support OSP
- 1.4.3 – appropriate for any OSP enabled device
- 2.1.1 – for devices which support version 2 OSP features
- 2.1.1-P – for proxy devices which support LookAhead routing

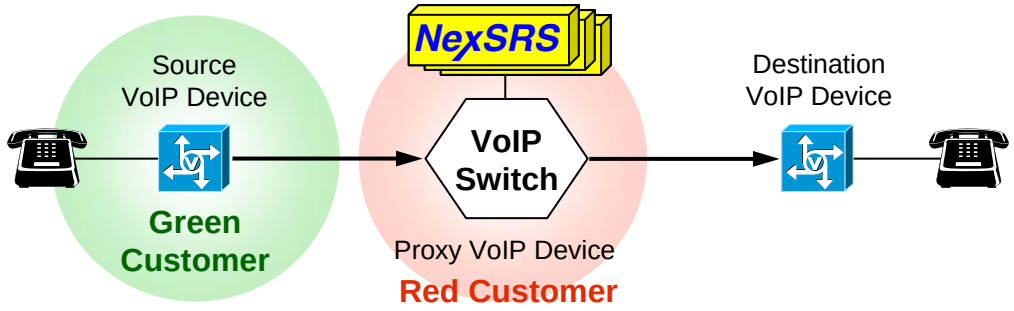
Strip In Tech Prefix – If this box is checked, the NexSRS server will always remove the In Tech prefix from the called number.

Subnet Status

May Terminate – If this status box is checked, the NexSRS will allow calls to be routed to the device. If this box is not checked, no calls will be routed to the device. The May Terminate box does not affect the May Originate status of a device. A device with the May Terminate box unchecked can originate VoIP calls.

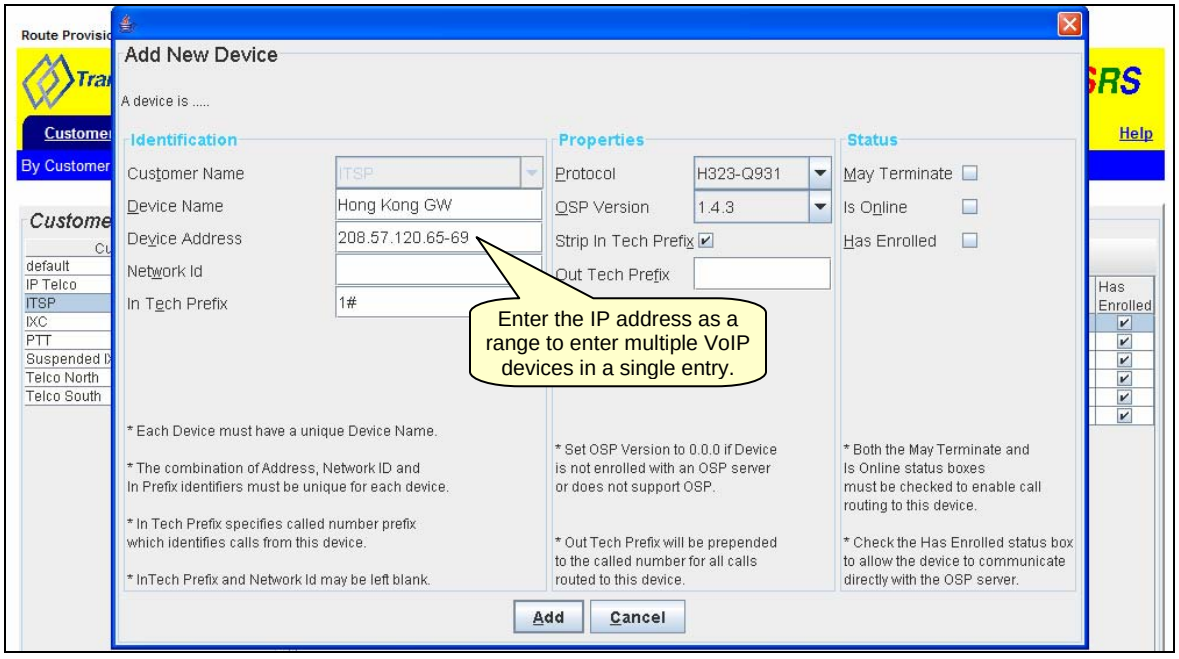
Is Online – The online status may be set manually by an operator or automatically by the device itself if the device is enabled to send OSP *CapabilitiesExchange* messages indicating the device is online or offline. Calls will not be routed to a device that has a status of offline.

Has Enrolled – This box must be checked for the device to send routing queries and call detail records directly to the NexSRS server. If this box is unchecked, call routing requests from the device will be denied unless the requests are sent via a proxy device (i.e. session border controller, IP to IP gateway, gatekeeper or SIP proxy) that has enrolled with the NexSRS server. For example, in the diagram below the proxy device must be enrolled with the NexSRS server since it communicate directly with the NexSRS for call routing and CDR reporting. The source VoIP device does not need to be enrolled since it does not communicate directly to the NexSRS server.

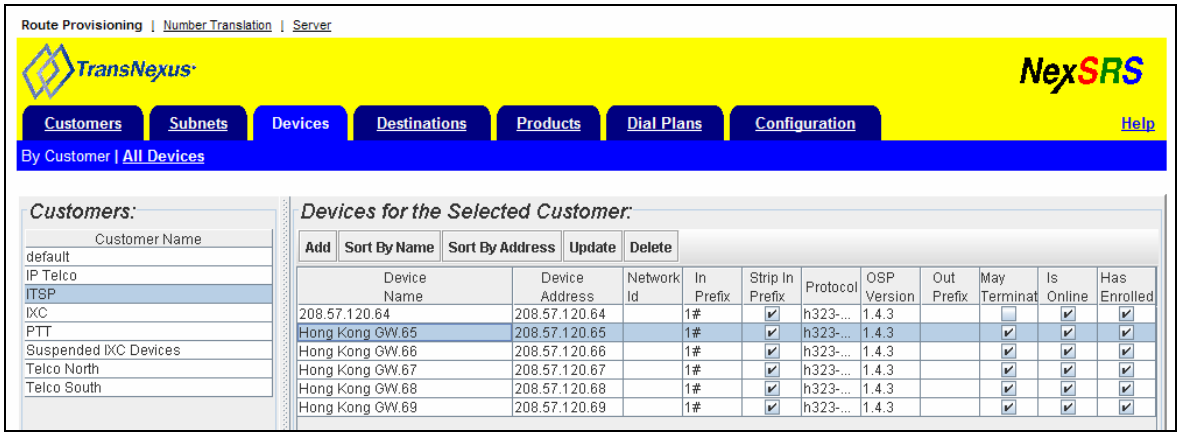


Adding multiple VoIP devices within an IP range

The NexSRS server provides a convenient shortcut for adding multiple VoIP devices with a continuous range of IP addresses. A single VoIP device can be added only once with the last octet of the IP addressed designated as a range as shown in the following screenshot.


Add New Device
A device is .....
Identification
Customer Name
ITSP
Device Name
Hong Kong GW
Device Address
208.57.120.65-69
Network Id
In Tech Prefix
1#
Properties
Protocol
H323-Q931
OSP Version
1.4.3
Strip In Tech Prefix
Out Tech Prefix
Status
May Terminate
Is Online
Has Enrolled
Enter the IP address as a range to enter multiple VoIP devices in a single entry.
* Each Device must have a unique Device Name.
* The combination of Address, Network ID and In Prefix identifiers must be unique for each device.
* In Tech Prefix specifies called number prefix which identifies calls from this device.
* InTech Prefix and Network Id may be left blank.
* Set OSP Version to 0.0.0 if Device is not enrolled with an OSP server or does not support OSP.
* Out Tech Prefix will be prepended to the called number for all calls routed to this device.
* Both the May Terminate and Is Online status boxes must be checked to enable call routing to this device.
* Check the Has Enrolled status box to allow the device to communicate directly with the OSP server.
Add
Cancel
Help
By Customer
default
IP Telco
ITSP
IXC
PTT
Telco North
Telco South
Has Enrolled

The NexSRS server will automatically populate the entry of multiple VoIP devices as shown in the following screenshot. The name of each VoIP device is appended with the last octet of the IP addresss to ensure all device names are unique.


Route Provisioning | Number Translation | Server
TransNexus
NexSRS
Customers
Subnets
Devices
Destinations
Products
Dial Plans
Configuration
Help
By Customer | All Devices
Customers:
Customer Name
default
IP Telco
ITSP
IXC
PTT
Suspended IXC Devices
Telco North
Telco South
Devices for the Selected Customer:
Add
Sort By Name
Sort By Address
Update
Delete
Device Name
Device Address
Network Id
In Prefix
Strip In Prefix
Protocol
OSP Version
Out Prefix
May Terminat
Is Online
Has Enrolled
208.57.120.64 208.57.120.64 1# h323-... 1.4.3
Hong Kong GW.65 208.57.120.65 1# h323-... 1.4.3
Hong Kong GW.66 208.57.120.66 1# h323-... 1.4.3
Hong Kong GW.67 208.57.120.67 1# h323-... 1.4.3
Hong Kong GW.68 208.57.120.68 1# h323-... 1.4.3
Hong Kong GW.69 208.57.120.69 1# h323-... 1.4.3

Chapter 5

Destinations, Products & Dial Plans

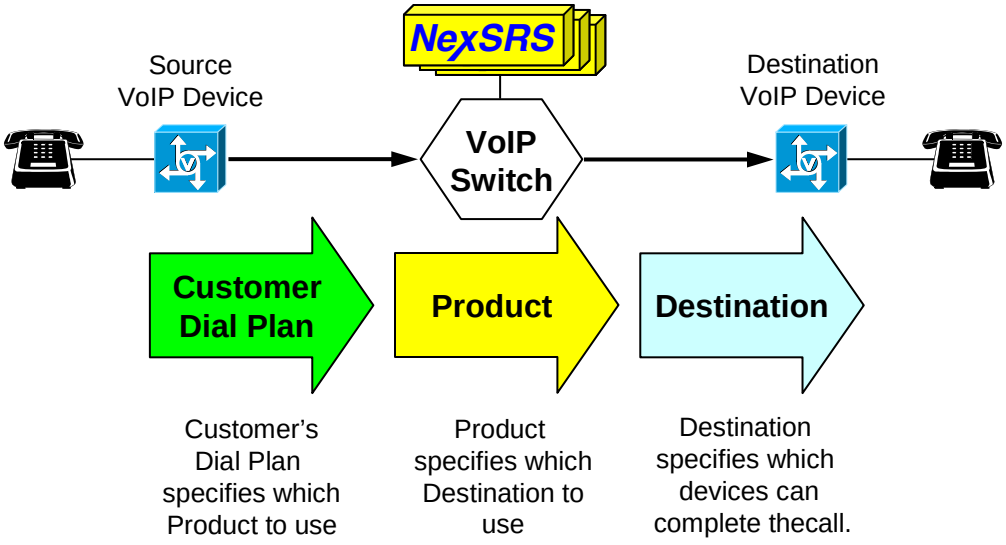
Introduction

The previous chapter defined how to provision Customers, Subnets and VoIP Devices – information which defines the network elements managed by the NexSRS server. This chapter introduces three new concepts which enable the NexSRS user with the flexibility to manage any VoIP traffic scenario. The concepts are:

Destinations. Destinations are groups of VoIP devices which terminate calls to a common set of Public Switched Telephone Numbers (PSTN).

Products. A Product is a set of routing services that is sold to Customers. A Product in NexSRS defines which Destination is selected to complete a call to the PSTN.

Dial Plans. A Dial Plan is created for each Customer and defines which Product is provided for a Customer for each call.



The diagram above illustrates the NexSRS routing algorithm. The Customer's Dial Plan configured in NexSRS defines which Product should be used to complete the call. The NexSRS Product defines which Destinations can complete the call. The Destinations define the specific VoIP Devices and IP addresses that should be returned in the routing response.

The routing algorithm logic flows from Dial Plan to Product to Destination. However, the provisioning steps are explained in this Chapter in reverse order. First, Destinations must be defined. Next, Products can be defined that route calls to Destinations. Last, Customer Dial Plans can be created that specify which Products should be used for different called numbers (breakouts).

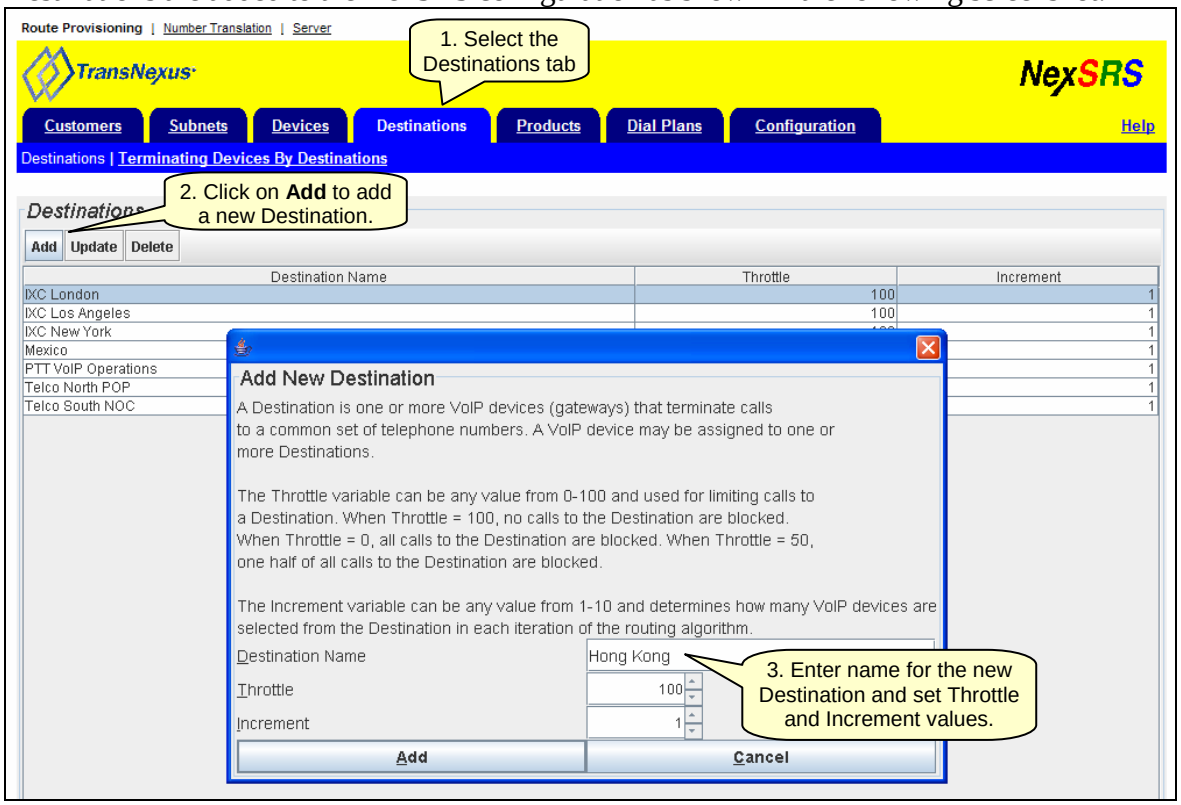
Breakouts

This chapter introduces the term breakout. A breakout is a dial string used for routing. Full telephone numbers are rarely used for routing. Instead, shorter numbers – breakouts – are used to define routing for large blocks of telephone numbers. For example, calls routed to TransNexus at 14045266060 would typically be routed according to the breakout 1404 which covers the local calling area in Atlanta.

Define Destinations

In the NexSRS peering server, VoIP calls are routed to Destinations. Destinations are defined as one or more VoIP devices that terminate calls to the same set of telephone numbers. A VoIP device may be assigned to one or more Destinations. VoIP devices from different Customers maybe included in the same Destination.

Destinations are added to the NexSRS configuration as shown in the following screenshot.


Route Provisioning | Number Translation | Server
1. Select the Destinations tab
TransNexus
NexSRS
Customers
Subnets
Devices
Destinations
Products
Dial Plans
Configuration
Help
Destinations | Terminating Devices By Destinations
2. Click on Add to add a new Destination.
Destinations
Add
Update
Delete
Destination Name
Throttle
Increment
IXC London 100 1
IXC Los Angeles 100 1
IXC New York 1
Mexico 1
PTT VoIP Operations 1
Telco North POP 1
Telco South NOC 1
Add New Destination
A Destination is one or more VoIP devices (gateways) that terminate calls to a common set of telephone numbers. A VoIP device may be assigned to one or more Destinations.
The Throttle variable can be any value from 0-100 and used for limiting calls to a Destination. When Throttle = 100, no calls to the Destination are blocked. When Throttle = 0, all calls to the Destination are blocked. When Throttle = 50, one half of all calls to the Destination are blocked.
The Increment variable can be any value from 1-10 and determines how many VoIP devices are selected from the Destination in each iteration of the routing algorithm.
Destination Name
Hong Kong
Throttle
100
Increment
1
3. Enter name for the new Destination and set Throttle and Increment values.
Add
Cancel

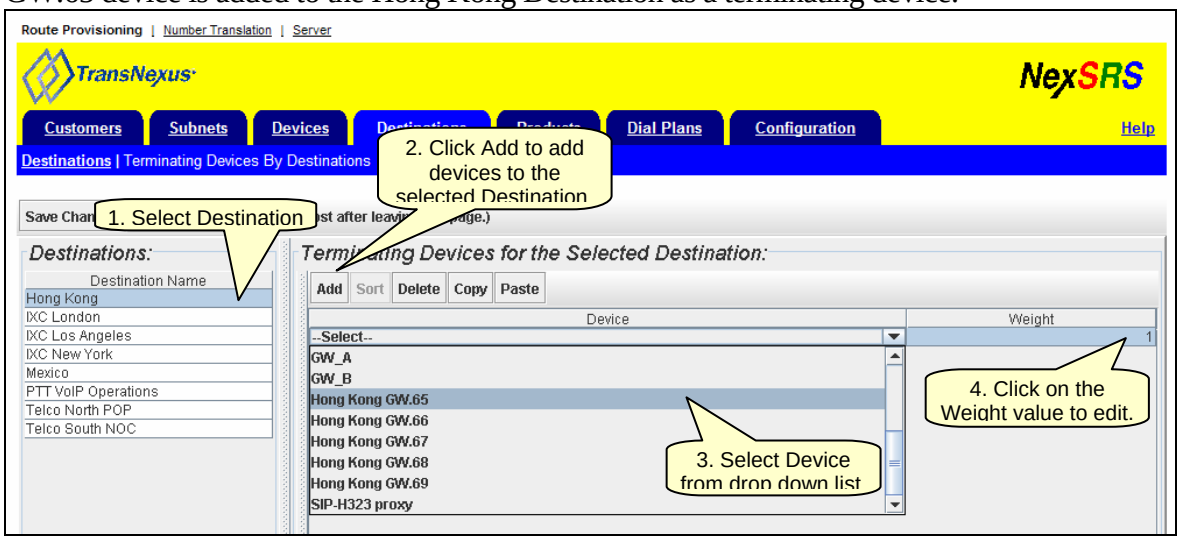
An important property for Destinations is the Throttle value. The Throttle value can be set to any integer from 0 to 100 and determines what percentage of calls will be routed to a

Destination. The default value is 100 – which results in 100% of all calls for a Destination being routed. A throttle value of 0 results in 0% of calls for a Destination being routed. In other words, a throttle value of 0 blocks all calls to a Destination. Throttle values between 0 and 100 represents partial blocking of calls which can be useful if a network operator needs to reduce call volumes to specific Destinations.

The Increment value may be set to any integer from 1 to 10. The default value is 1. The Increment value is used be the NexSRS routing algorithm when multiple Destinations can terminate the same call. For each routing algorithm iteration, the Increment value determines how many VoIP devices are taken from each possible Destination and added to the final list of VoIP devices. The Increment value should always be less than or equal to the number of Devices assigned to a Destination. TransNexus recommends that Increment be set to 1.

Assigning Devices to Destinations

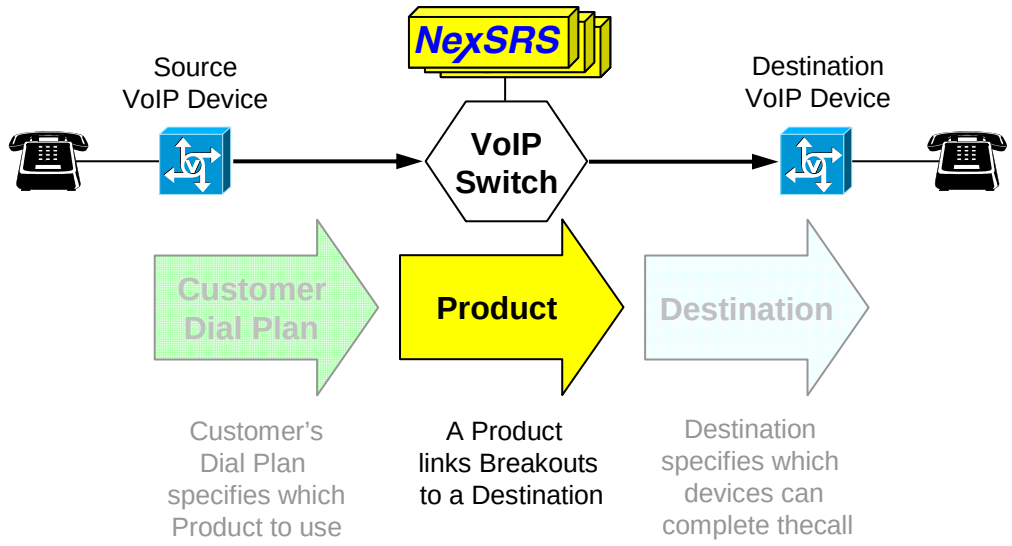
The second step in defining Destinations is to add terminating VoIP devices. Zero, one or more VoIP Devices may be added to a Destination. This screenshot illustrates how the Hong Kong GW.65 device is added to the Hong Kong Destination as a terminating device.



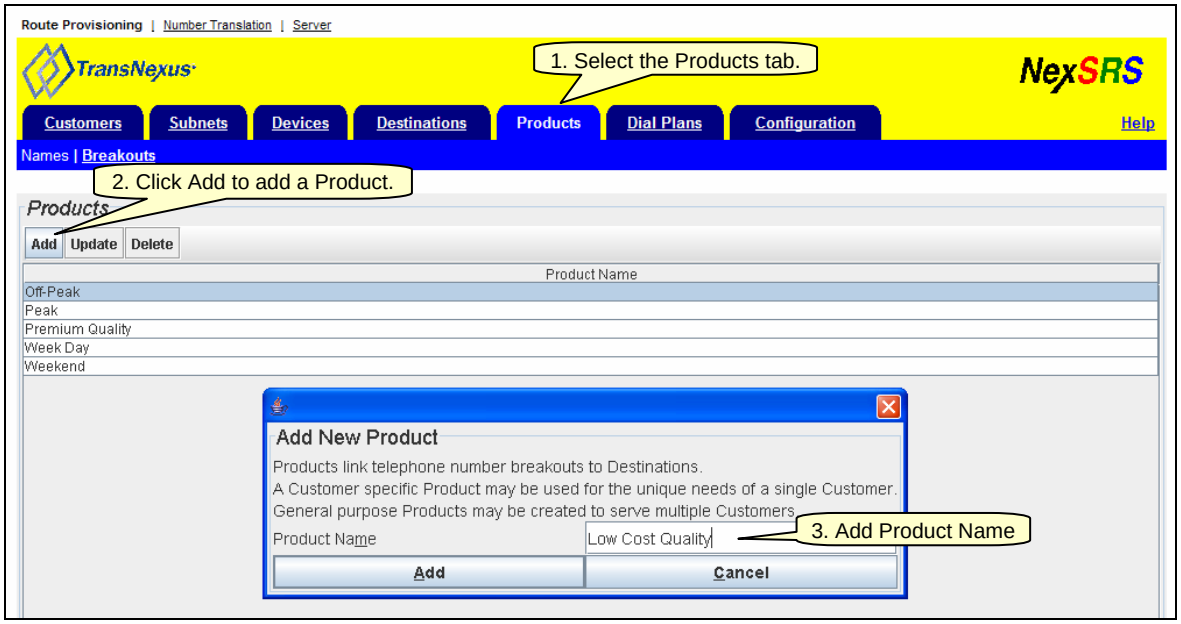
The Weight for each device is used for load balancing devices within the Destination. Devices with equal weights will share termination traffic equally. Devices with a greater weight are more likely to be selected for call termination. For example, if a Destination has Device-A with a weight of 2 and Device-B with a weight of 1, Device-A is twice as likely as Device-B to be selected as the first route choice. The default value for the weight parameter is 1 and the weight parameter can be edited directly on the web user interface.

Create Products

After Destinations have been defined, the next step in route provisioning is to create VoIP Products that will be provided to Customers. A Product links telephone number breakouts to Destinations as highlighted in the diagram below.



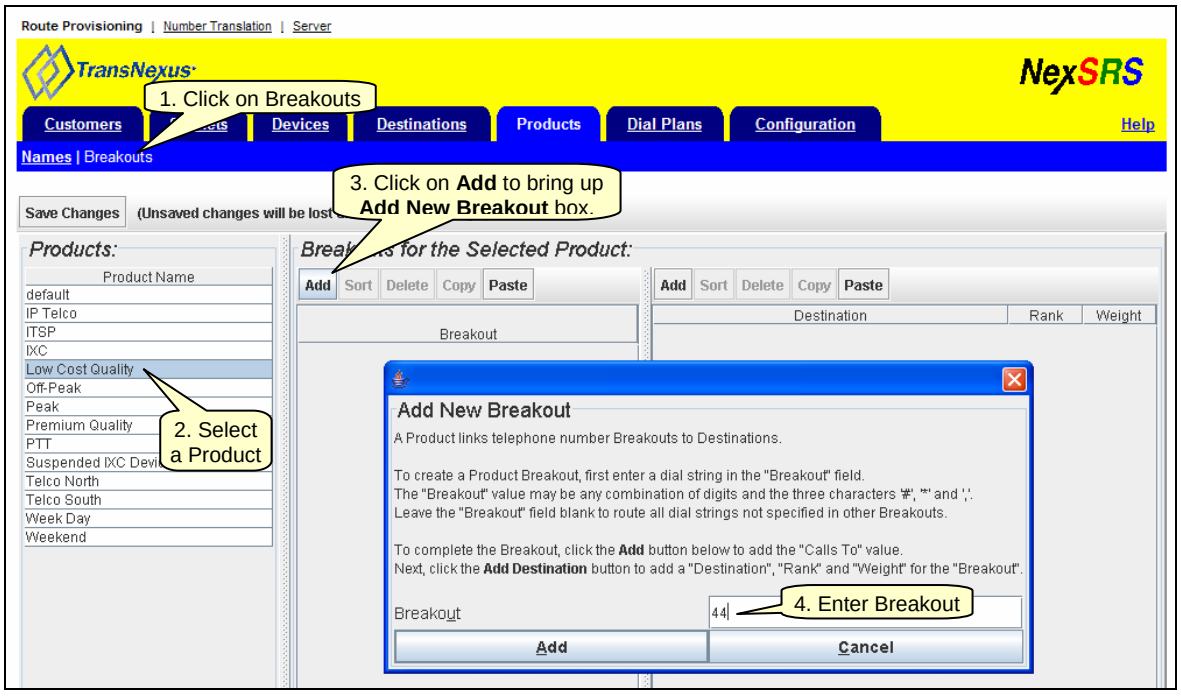
The first step to create a Product in the NexSRS server is to add the Product name as shown in the screen shot below.



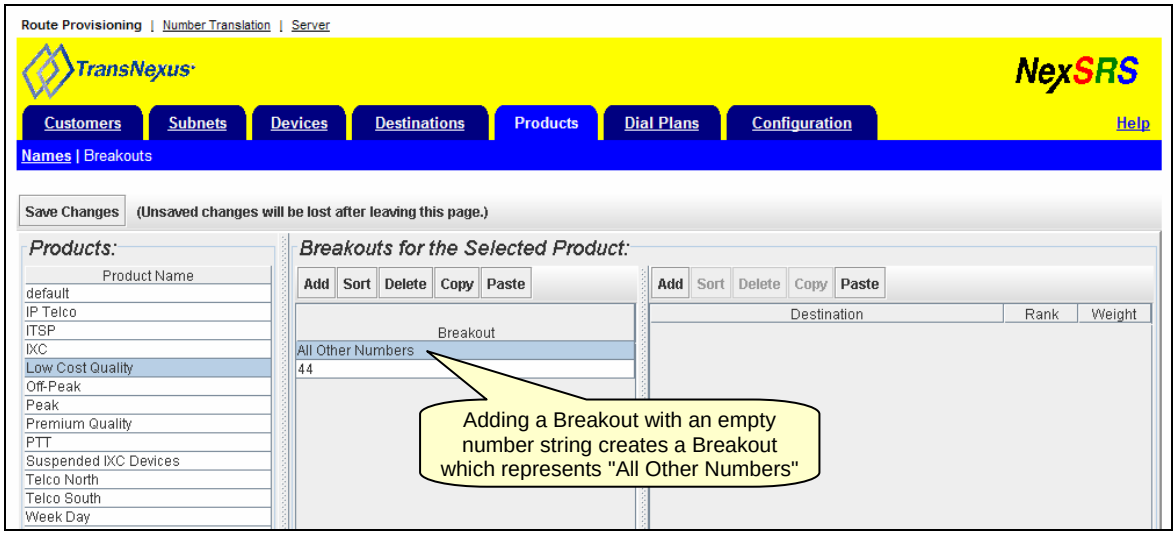
Configure Products

To configure a Product, as shown in the following screenshot, first select the Breakouts link, second select the Product that you want to configure. Note, that in addition to Products which have been created, such as "Low Cost Quality", there are also Customer specific Products.

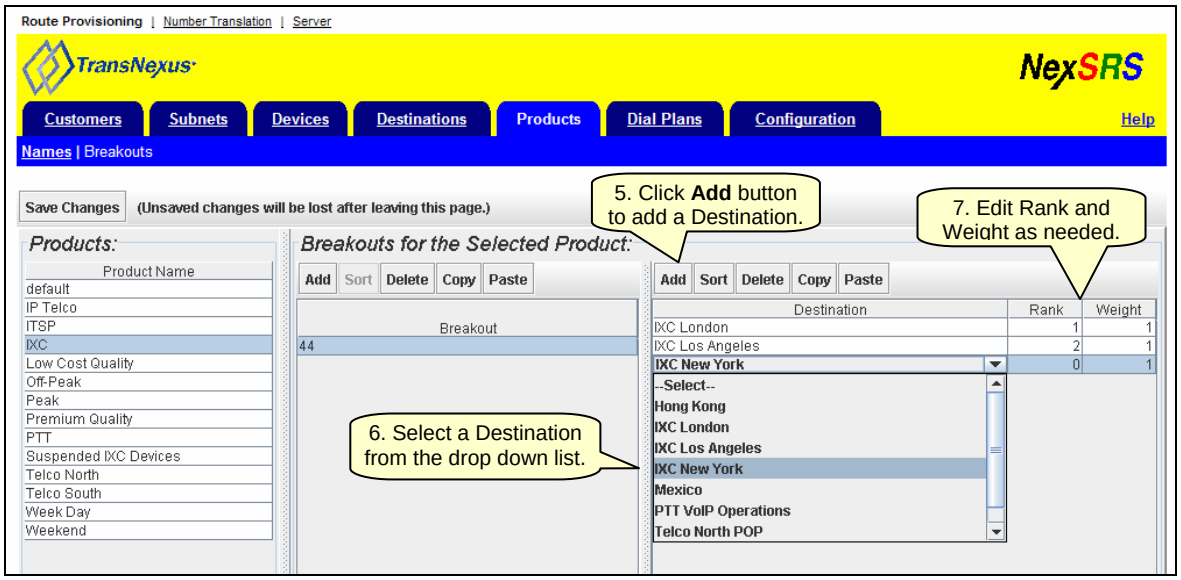
Customer specific Products are created automatically when a Customer is added to the NexSRS server. Customer specific Products may be used to create a unique Product for a single customer. The third step is to add a breakout for the selected Product. In the screen shot below, a new breakout for country code 44 is being created for the "Low Cost Quality" Product.



A telephone number breakout can be any string of numbers 0-9 and the three non-numeric digits: hash mark (#), asterisk (*) and comma (,). Typically, a breakout is an E.164 compliant dial string beginning with the country code. However, the non-numeric digits may be used to indicate trunk groups, technical prefixes or other number translations. If no number is entered in the **Add New Breakout** box, the breakout will represent all breakouts not configured, or All Other Numbers as shown in the following screenshot.



The combination of a Breakout and one or more Destinations creates a route in a Product. To complete the route in a Product, one or more Destinations may be configured for each Breakout. If no Destination is entered for a Breakout, calls for the route will be blocked. Click the **Add** button, above the Destination column header, to add a Destination as shown in the following screenshot. Up to twelve Destinations may be added per route. For each Destination, specify its rank for routing priority. Destinations with Rank=1 are first priority. When multiple Destinations have the same Rank, the weight parameter is used for load balancing. Destinations with a greater weight are more likely to be the selected Destination compared to other Destinations with the same rank and lower weights.



Rank Ordering and Load Balancing with Weights

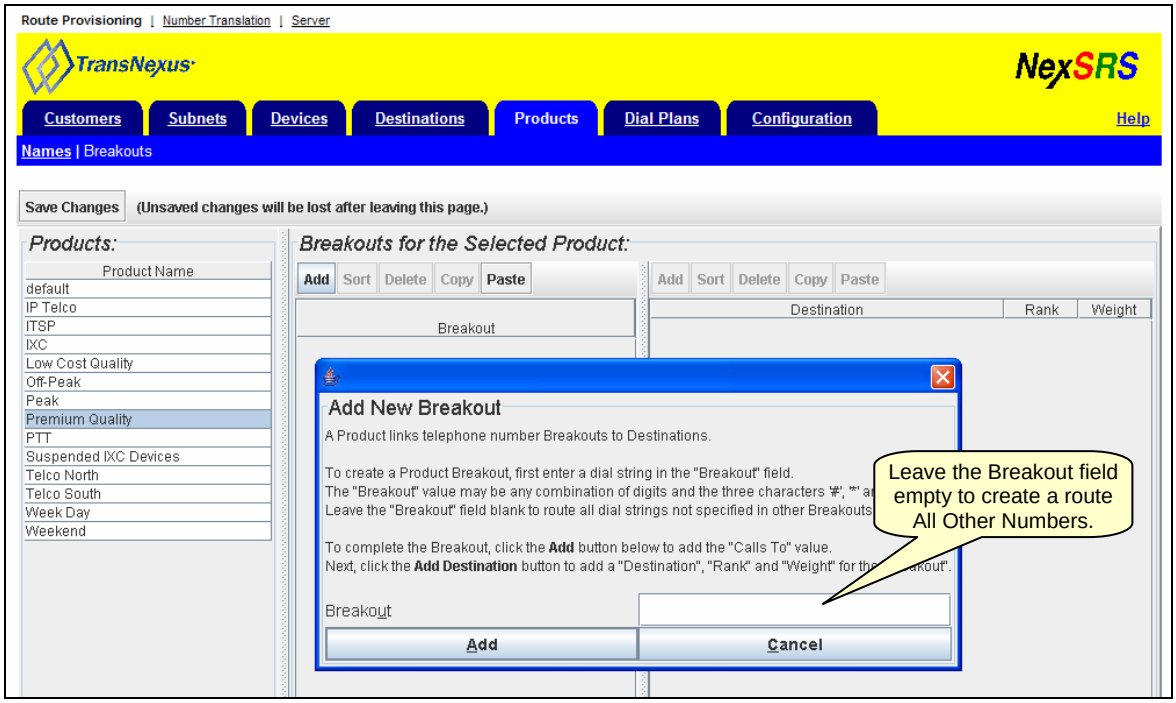
For each route request, the NexSRS server generates a list of Destinations based on the rank order of Destinations. A Destination with Rank=1 will always be first. Destination ranks do not have to be sequential. A Destination with Rank=37 gets priority over a Destination with Rank=100. Rank ordering is a convenient tool to configure Destinations according to Least Cost Routing (LCR).

Rank=0 is a special case. A Destination with Rank=0 is always the backup Destination in a routing query response. If the number of possible Destinations is too many to include in a routing query response, a VoIP device from a Destination with Rank=0 will always be included as the last terminating device in the routing response returned to the source VoIP device. Typically, the most reliable (and expensive) Destination is configured with Rank=0.

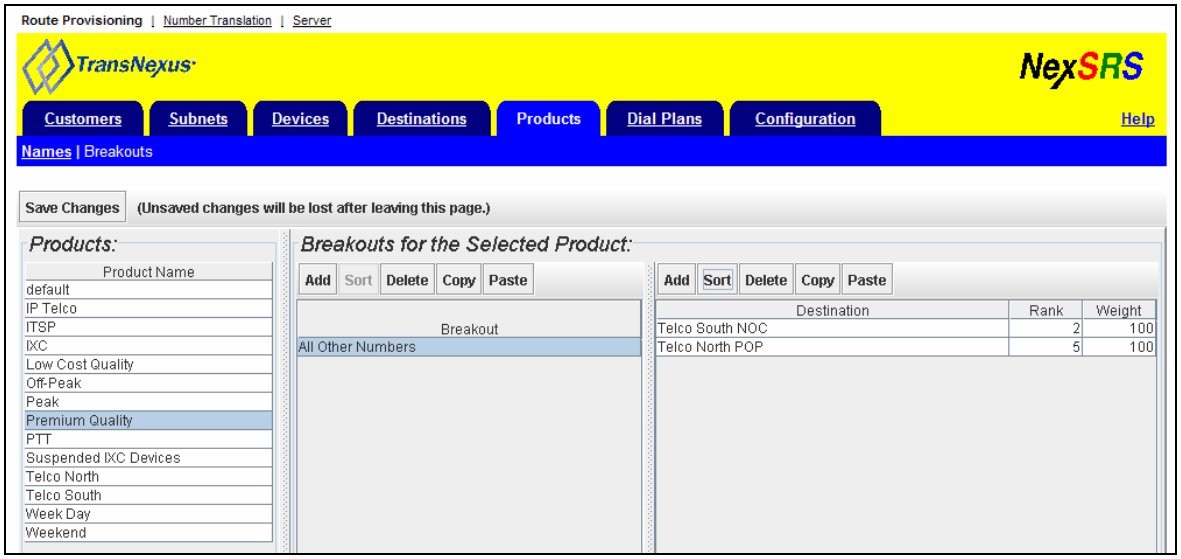
When one or more Destinations have the same rank, the Destinations are randomly load balanced according to their weights. Destinations with a higher weight are more likely to be returned as a route than a Destination with a lower weight. Please see Appendix 2 for a detailed description of the load balancing algorithm.

Global Routing

Global routing is achieved by not specifying a dial string for a Breakout. This is done by clicking on the **Add** button to open the **Add New Route** pop-up box as shown below. In the **Add New Route** pop-up, leave the Breakout dial string box empty to represent all numbers not explicitly configured as routes. An empty Breakou will create a Breakout for "All Other Numbers."

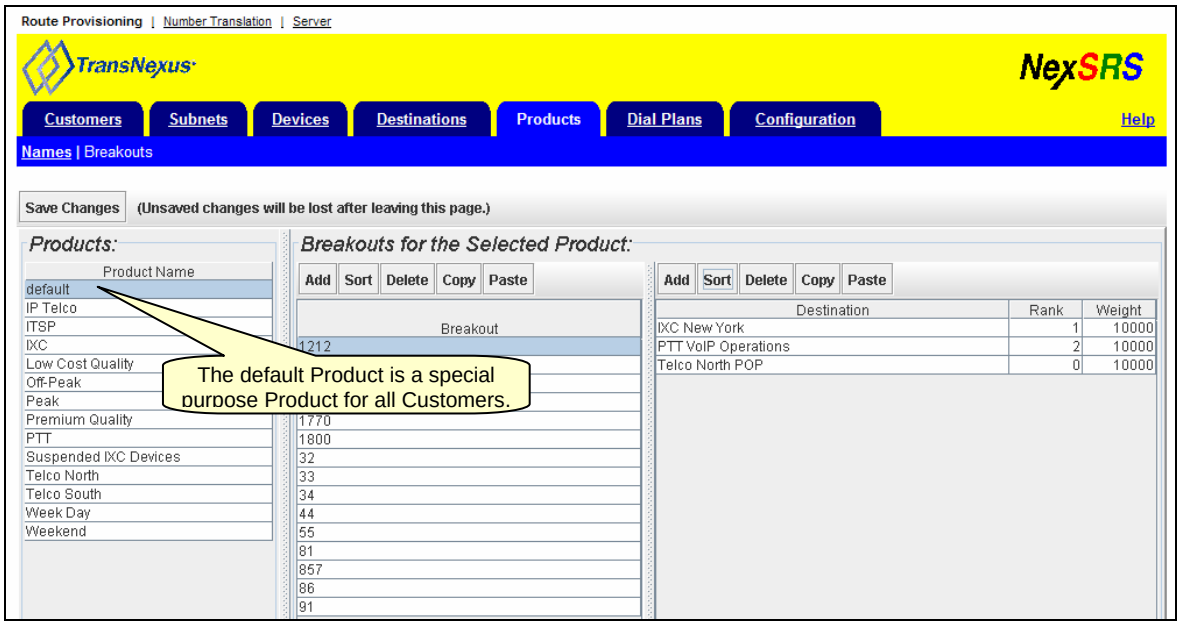


The screenshot below shows a route to All Other Numbers for the "Premium Quality" Product. Traffic to all other numbers will be route first to the Telco South NOC since it has higher rank than the Telco North POP.



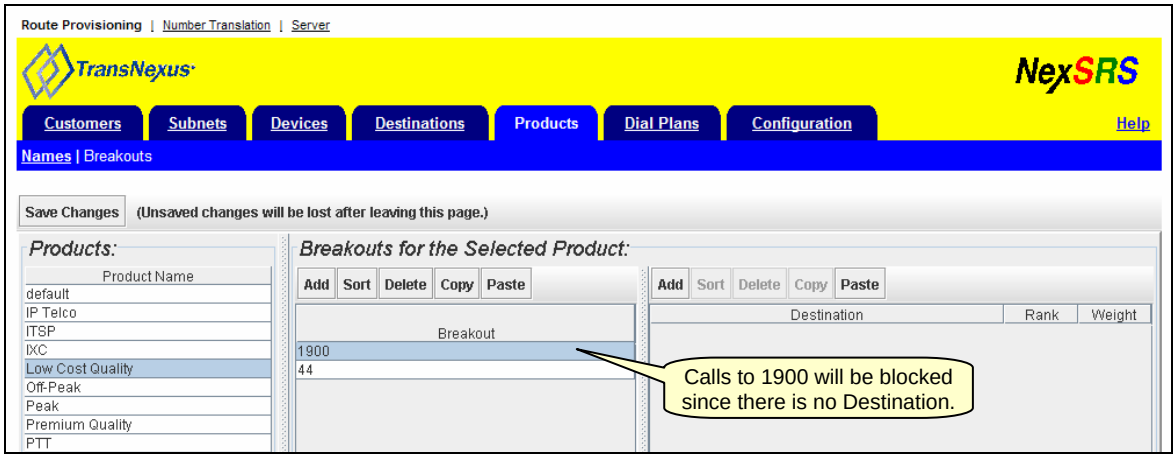
Global Product

A global Product for all Customers can be created by using the special purpose Product name default (with a lower case d). Below is a screen shot of the default Product configured with routes.



Blocking Calls to Specific Dialed Numbers

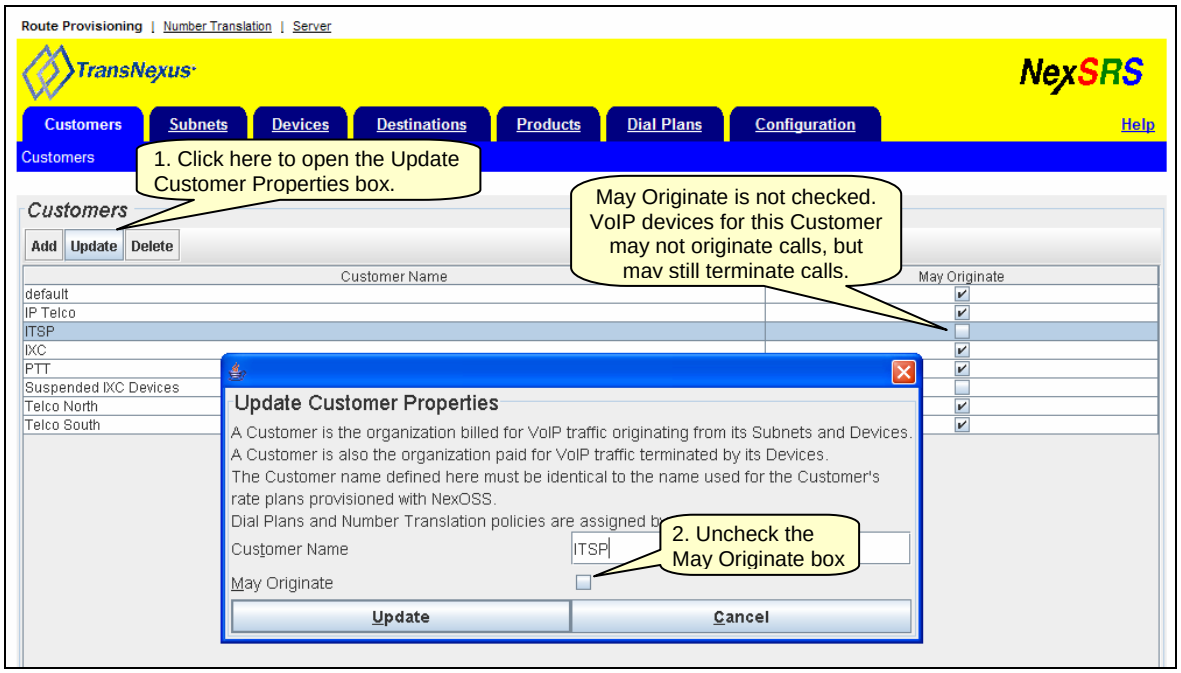
If no Destinations are assigned to a Breakout, then all calls to the Breakout will be blocked. This feature is a simple way to block calls in a Product to an un-authorized Breakout. In the following example, no Destinations have been entered for the Breakout 1900 of the "Low Cost Quality" Product. The "Low Cost Quality" Product will block all calls beginning with 1900.



Other Routing Features

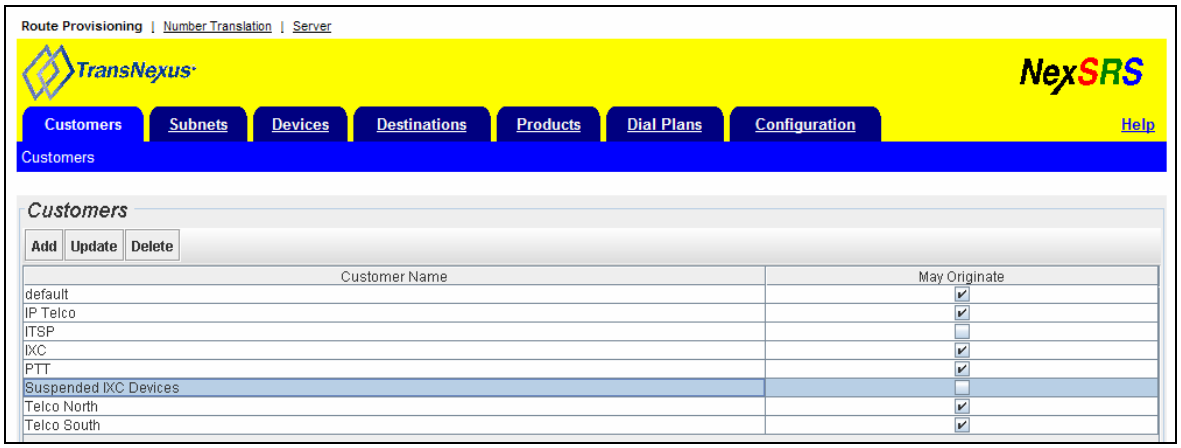
Blocking All Calls from a Customer

Business policies may require that a Customer can terminate calls, but may not originate calls. To enforce this policy, Select the Update button on the Customers page and uncheck the **May Originate** box for the Customer.

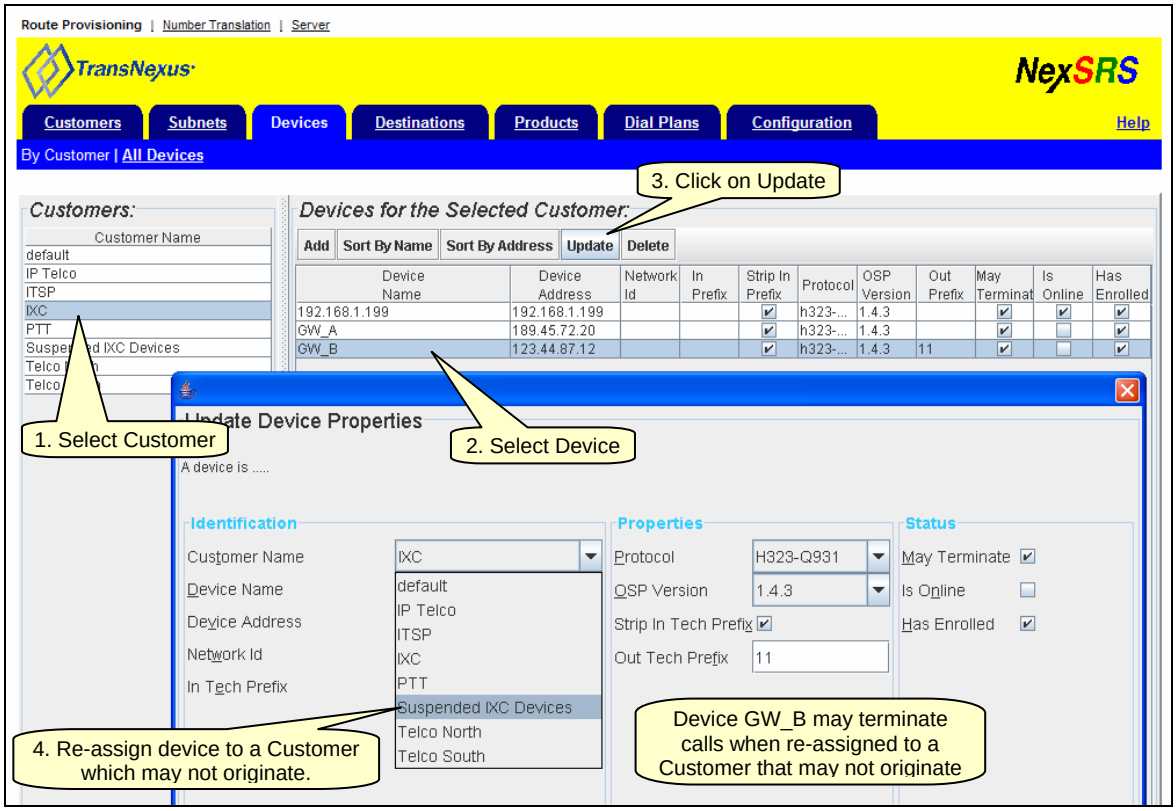


Blocking Calls from a VoIP Device

To block calls from a specific Customer VoIP device, without impacting the routing of calls from other Customer VoIP devices, reassign the device to a new Customer that is not allowed to originate calls. In the example below, a new Customer "Suspended IXC Devices" has been created for devices from the IXC Customer which should not be allowed to originate calls, but which may terminate calls.



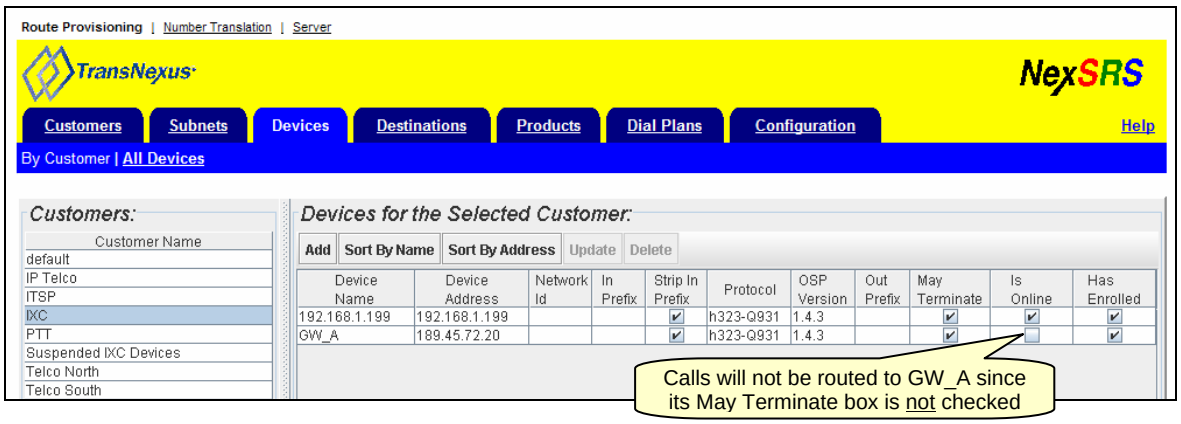
The May Originate box of the "Suspended IXC Devices" is not checked, so VoIP devices in assigned to this Customer may not originate calls. However, the routing status of devices for this Customer is not changed for terminating calls. The screenshot below illustrates the steps to re-assign a device from one Customer to another.



To re-enable call origination privileges for the termination device in the example above, the device must be re-assigned to the IXC Customer which may originate calls.

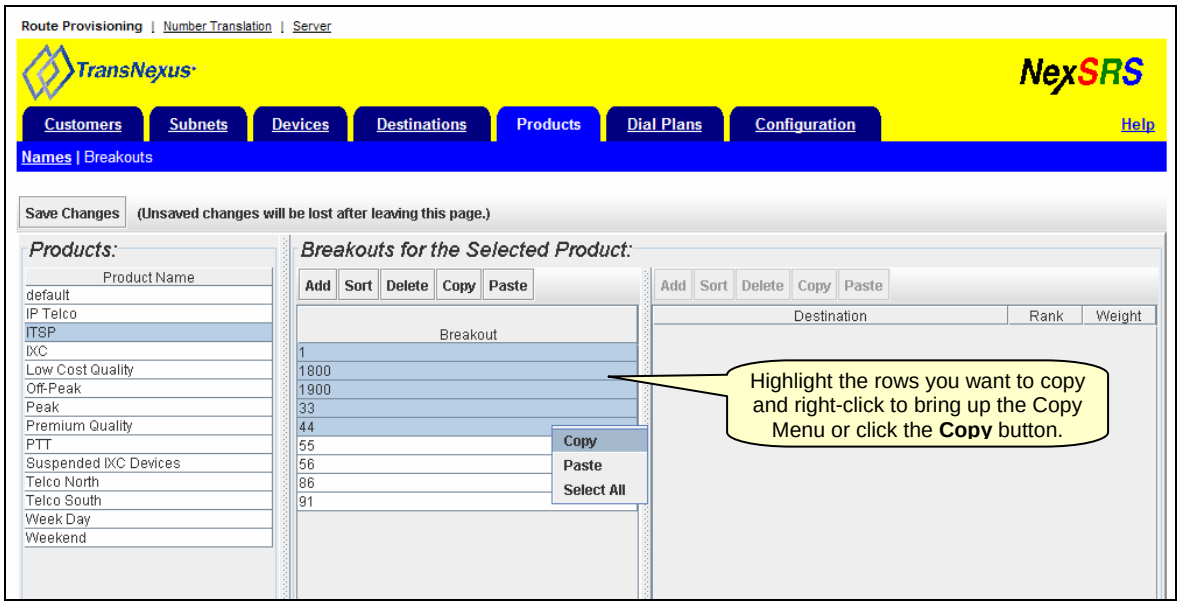
Blocking Calls to a VoIP Device

To block calls to a specific device, uncheck the May terminate box for the device. For example, in the screenshot below, the May terminate box for GW_A is not checked. No calls will be routed to GW_A since it may not terminate calls.



Editing Routes

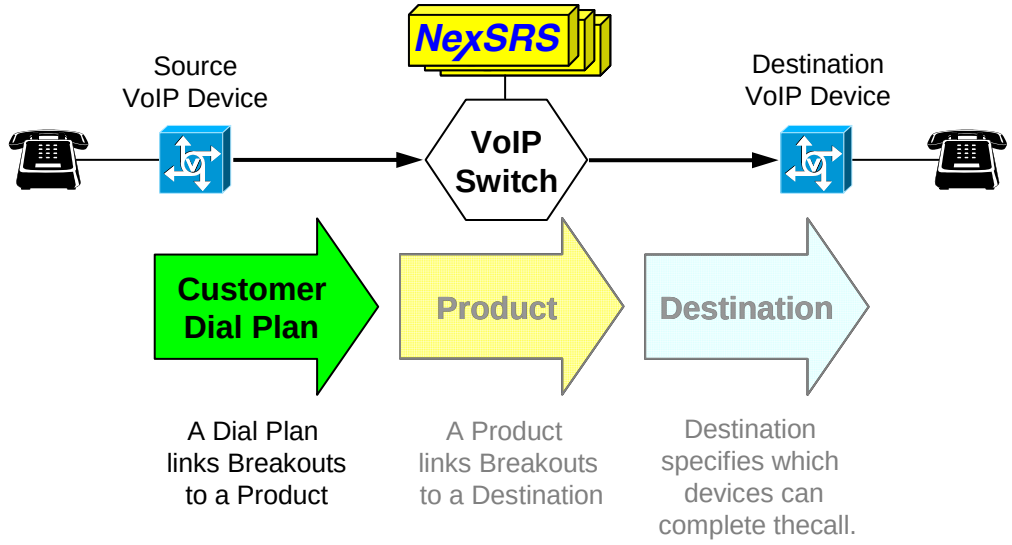
Copy and paste functions enable easy re-provisioning of routes among different Products. To copy multiple routes from one Product to another, select the routes in the Breakout column and right click to bring up the copy/paste menu as shown in the screen shot below.



Next, select the Product to which you want to paste routes and use the Paste button to paste the routes. The routing table of the NexSRS server may also be edited manually using any text editor. See Appendix 3 for a description of the nexus.cfg file.

Dial Plans

After Products are created, the next step is to create Dial Plans for each Customer. Dial Plans link Breakouts to Products. The Customer's Dial Plan specifies which Product is used by Breakout.



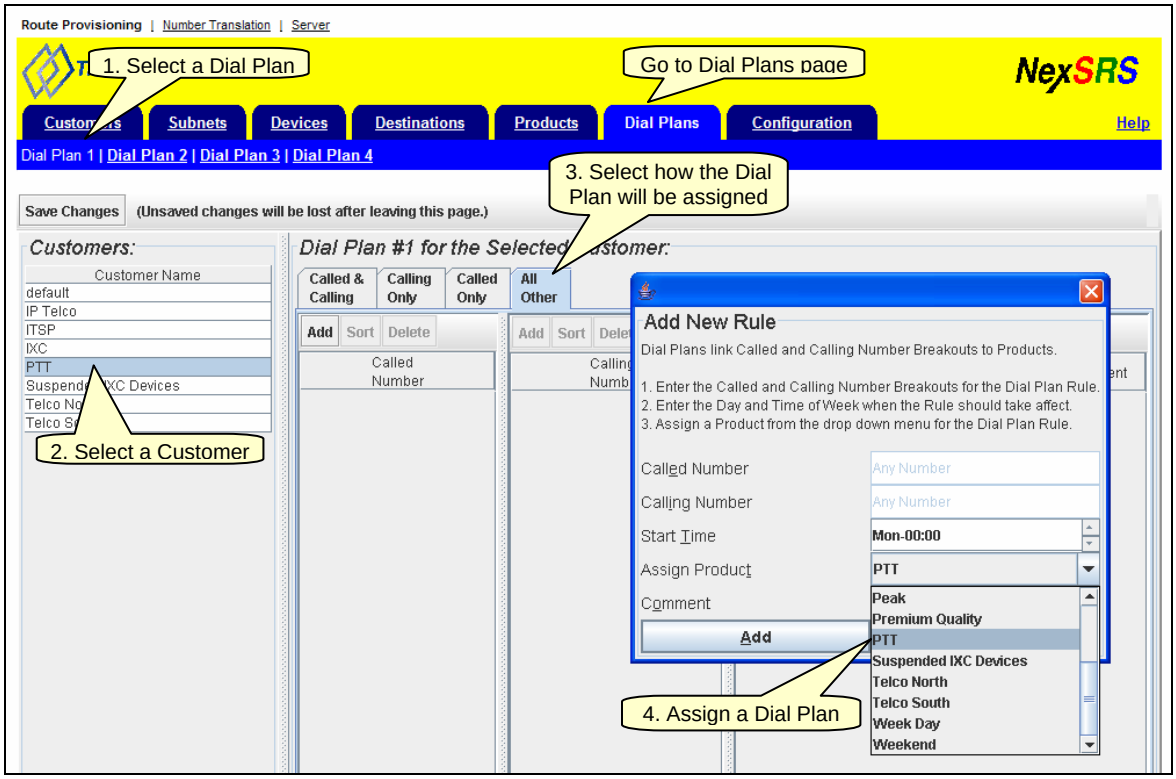
For example, A Dial Plan may specifiy that calls to Breakout 44 use the "Low Cost Quality" Product and calls to breakout 81 use the "Premium Quality" Product. Up to four Dial Plans may be assigned for each Customer. In a Dial Plan, Products are assigned based on any combination of calling and called number. The NexSRS server will sequentially search a Customer's Dial Plans until a Product assignment is found that matches the calling and called number combination.

How to Create a Dial Plan

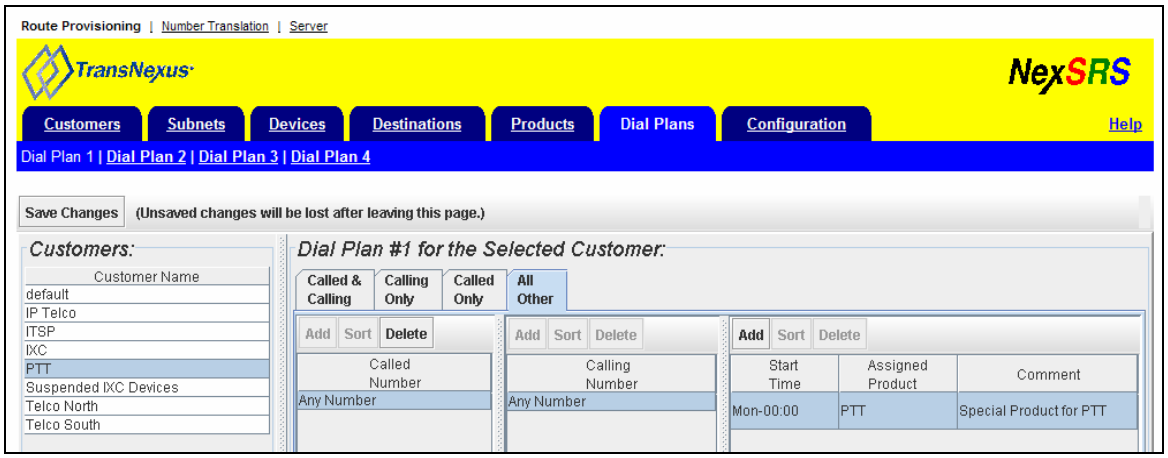
The following examples illustrate how to create Dial Plans. First, select the Dial Plan 1 link on the Dial Plans page. Second, select a Customer. Third, select the Dial Plan tab which describes how the Product will be assigned. The choices are:

Dial Plan Tab	Description
Called and Calling	Product is assigned by the Called and Calling number
Calling Only	Product is assigned by the Calling number
Called Only	Product is assigned by the Called number
All Other	Product is assigned by any Calling or Called number

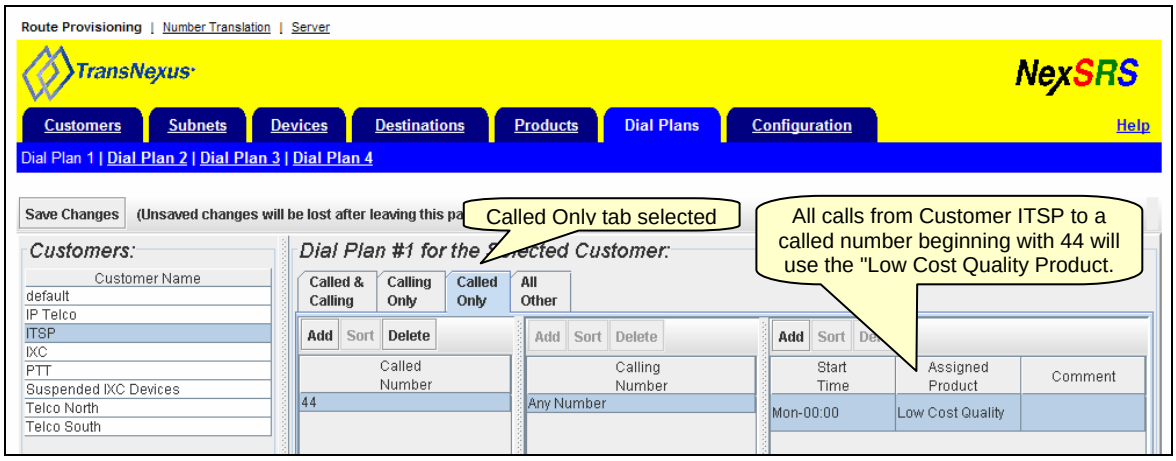
In the following example, the **All Other** Dial Plan tab has been selected. This means that the Product is assigned for any calling or called number. The fourth step is to click on the **Add** button to bring up the **Add New Rule** box and assign a Product from the drop down menu. Click on the **Save Changes** button when finished – otherwise the changes will not be saved..



The results from the previous examples are shown in the following screenshot. In Dial Plan 1 for the PTT Customer, the PTT Product has been assigned for all calls (any calling or called number).

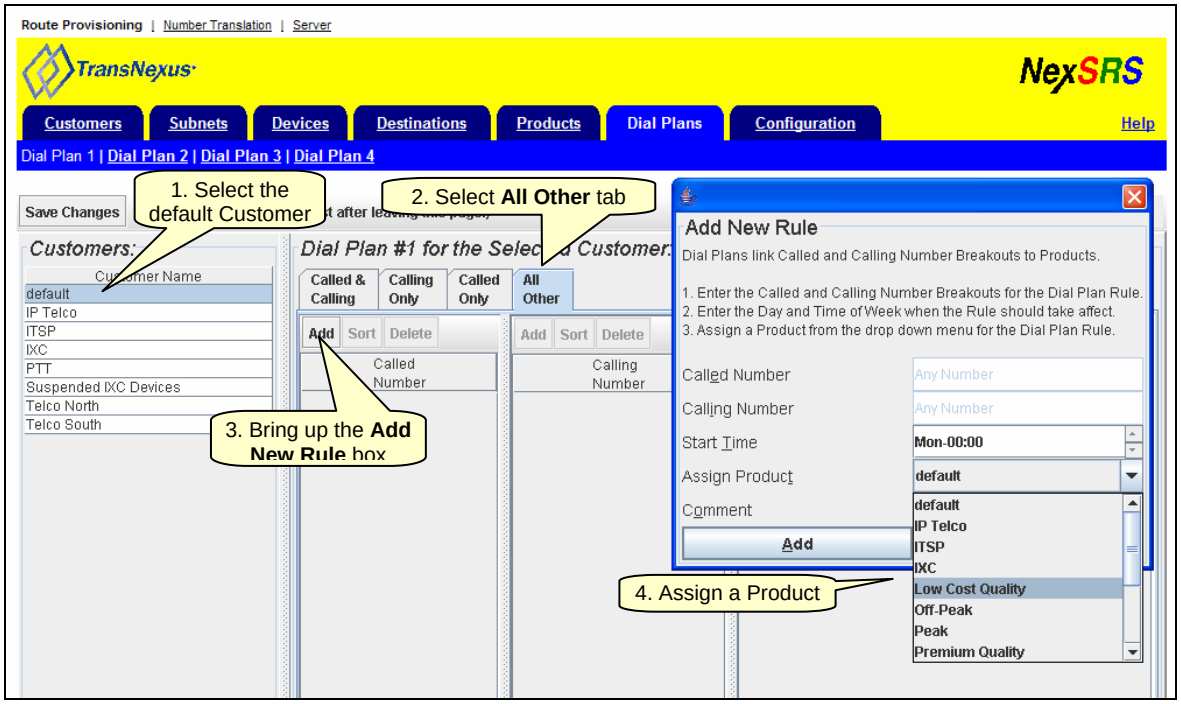


The following example illustrates a Product assigned by the called number. Dial Plan 1 is configured for the ITSP customer. The Product "Low Cost Quality" is assigned based on the Called Number Breakout 44. All calls to telephone numbers beginning with 44 will be routed according the 'Low Cost Quality" Product.



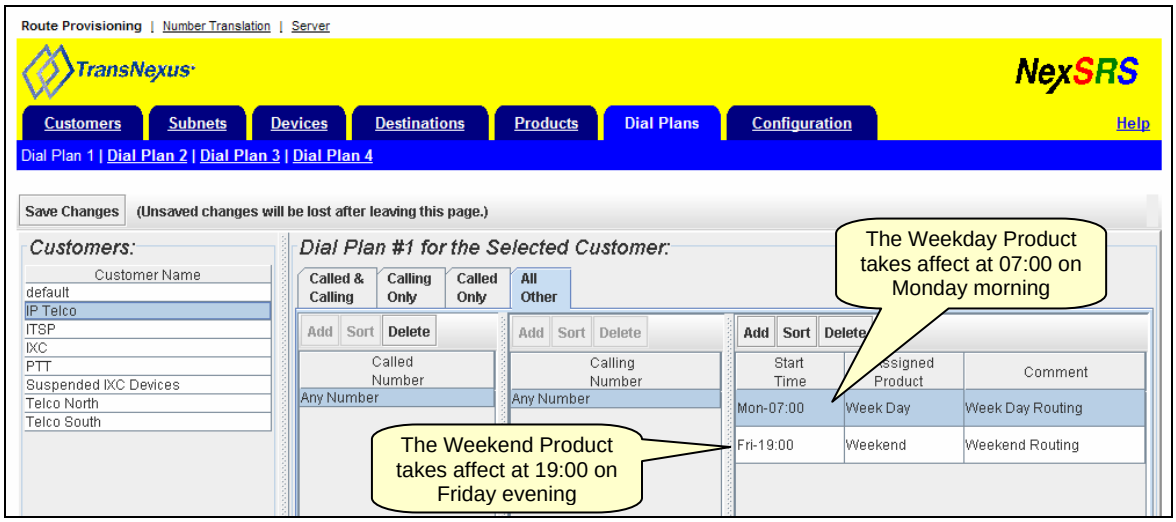
Global Dial Plan

The simplest and quickest way to provision service for all Customers is to create a Dial Plan 1 for the default (lower case d) Customer and assign a Product for the **All Other** tab (any calling or called number). The Product assigned for the default Customer will apply for all calls from all Customers unless superceded by a Customer specific Dial Plan. The steps for assigning a Dial Plan for the default Customer are shown the in the following screenshot.



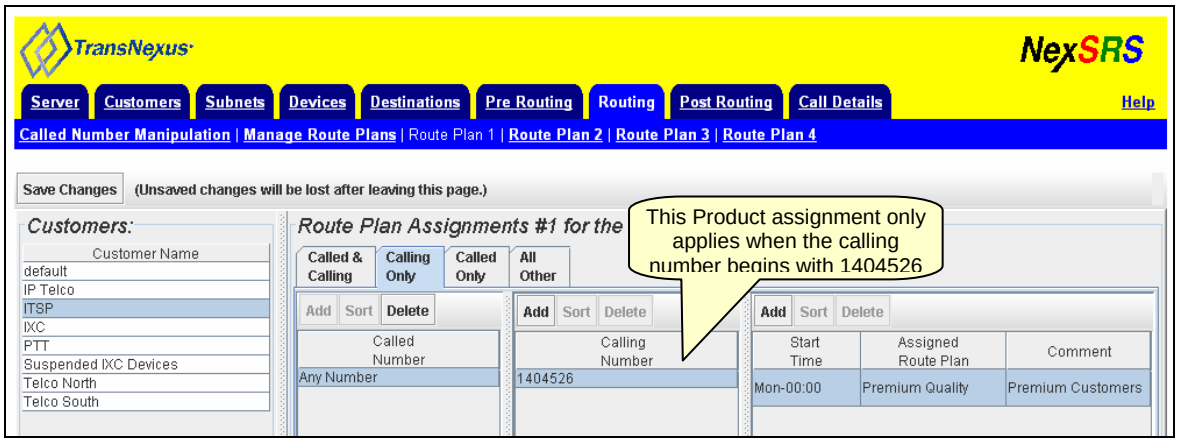
Day of Week Routing.

The screenshot below provides an example of how Day of Week routing can be implemented. In this example, the "Week Day" Product is assigned to take effect at 07:00 on Monday morning and stay in effect until the Weekend Product takes effect at 19:00 on Friday evening. The Weekend Product remains in effect until 07:00 Monday morning when it is replaced by the Weekday Product and the cycle repeats.



Routing Based on Calling Number.

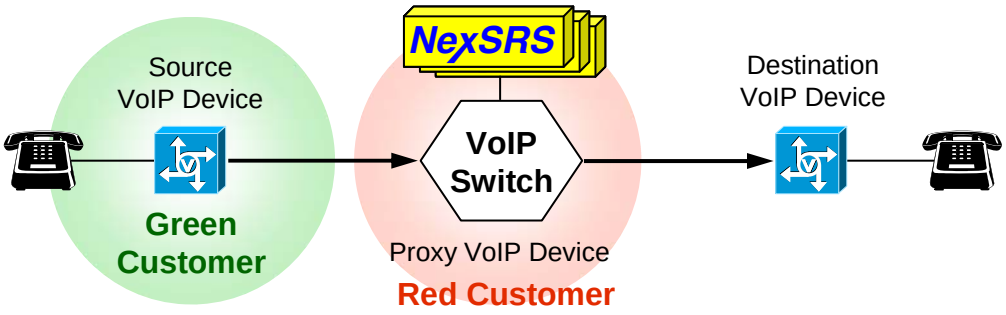
The following screenshot is an example of ANI based routing, or routing based on the Calling Number. In the following screenshot the "Premium Quality" Product has been assigned in Dial Plan 1 for Customer ITSP when the Calling Number begins with 1404526.



It is important for the NexSRS user to understand that the Product assignment above is limited to calls with Calling Number beginning with 1404526. To provide services for calls with other Calling Numbers, additional breakouts need to be configured for the Dial Plan.

Dial Plan Selection Algorithm

It is important for the NexSRS operator to understand the Dial Plan selection algorithm. The NexSRS server searches through forty-eight possible Dial Plan choices to find the correct Dial Plan for each call. When The NexSRS server receives a routing query, the first step in determining routing policy is to determine the source device of the call and to which Customer the device is assigned since Dial Plans are configured by Customer,



For example, consider the diagram above, a VoIP call originates from the source device in the Green Customer's network. The call proceeds to the VoIP Switch in the Red Customer's network. The VoIP switch sends the routing query to the NexSRS server that responds with routing address of the destination device. In this example, the source device belonging to the Green Customer is the source of the call. The NexSRS server therefore would first search for a Dial Plan assigned to the Green Customer. The Dial Plan selection process begins by searching Dial Plan 1 of the Green Customer. The NexSRS first searches for a match between the calling and called numbers and breakout configured under the **Called & Calling** tab. If a match is found,

the assigned Product is used to determine the call route. If no match is found, the NexSRS server then searches for a match between the calling number and breakouts configured under the **Calling Only** tab. If a match is found the assigned Product is used to route the call. If no match is found, the NexSRS server searches for a match between the called number and the **Called Only** tab. If a match is found the assigned Product is used to route the call. If no match is found, the NexSRS server searches for a Product assigned under the **All Other** tab. If no match is found, the NexSRS server repeats the process for Dial Plans 2, 3 and 4 of the Green Customer.

If no match is found in Dial Plans configured for the Customer of the source device, the NexSRS will next search the Dial Plans configured for the Customer of the proxy device. For example, in the previous diagram, the VoIP Switch is the proxy device routing the call from the source to the destination. The VoIP Switch is assigned to the Red Customer. So, if no Product assignment is found in the Dial Plans of the Green Customer, the NexSRS will repeat the search process using the Dial Plans of the Red Customer.

If the NexSRS does not find a Product assigned in the Dial Plans of the Customer of the source device (Green Customer) or in the Dial Plans of the proxy device (Red Customer), the last iteration is to search the Dial Plans of the default Customer. If no Product assignment for the call is found in the default Customer, the NexSRS returns – No Route Found – to the device which sent the route query. Please see Appendix 3 for more detailed information on the Dial Plan selection algorithm.

Suggested Routes

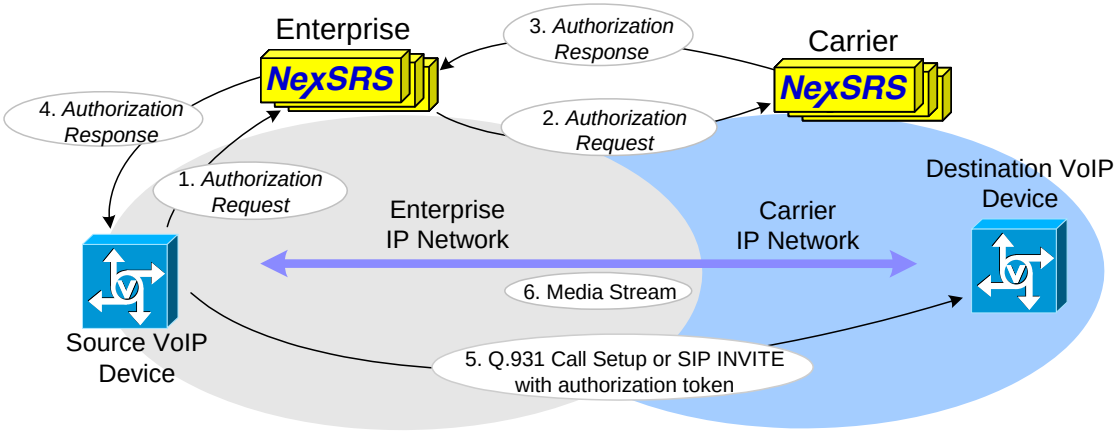
The OSP peering protocol includes the functionality that allows a source VoIP device to include one or more suggested termination devices in its OSP peering *AuthorizationRequest*. The purpose of the suggested routes feature is to enable the Source VoIP device to determine its own route decision, but still use the NexSRS server for secure inter-domain authorization and CDR collection. When one or more suggested routes is included in the OSP *AuthorizationRequest*, the NexSRS server will authenticate the source device as usual and then determine if the suggested destination VoIP devices are enabled in the NexSRS server configuration. If the destination devices are enabled, the NexSRS Server will respond to the source device with a digitally signed authorization token for each destination device. The NexSRS Server routing table is not used if a route is suggested by the source VoIP device.

Suggested routing by source VoIP devices is accomplished by including the IP address of the destination VoIP device in the OSP *DestinationAlternate type="transport"* field of the OSP *AuthorizationRequest*. The NexSRS Server can accept multiple suggested destination VoIP devices in an OSP *AuthorizationRequest*. Please refer to the documentation of your source VoIP device to determine if it supports suggested routing and how to configure the source VoIP to include suggested routes in an OSP *AuthorizationRequest*.

ExtraNet Routing

The NexSRS server can be configured to query other OSP servers for routing information when a route is not available internally. This feature, known as ExtraNet routing, enables an

enterprise operator to manage VoIP traffic among enterprise VoIP devices and leverage the VoIP termination of its carriers VoIP network. The diagram below illustrates an ExtraNet call scenario.



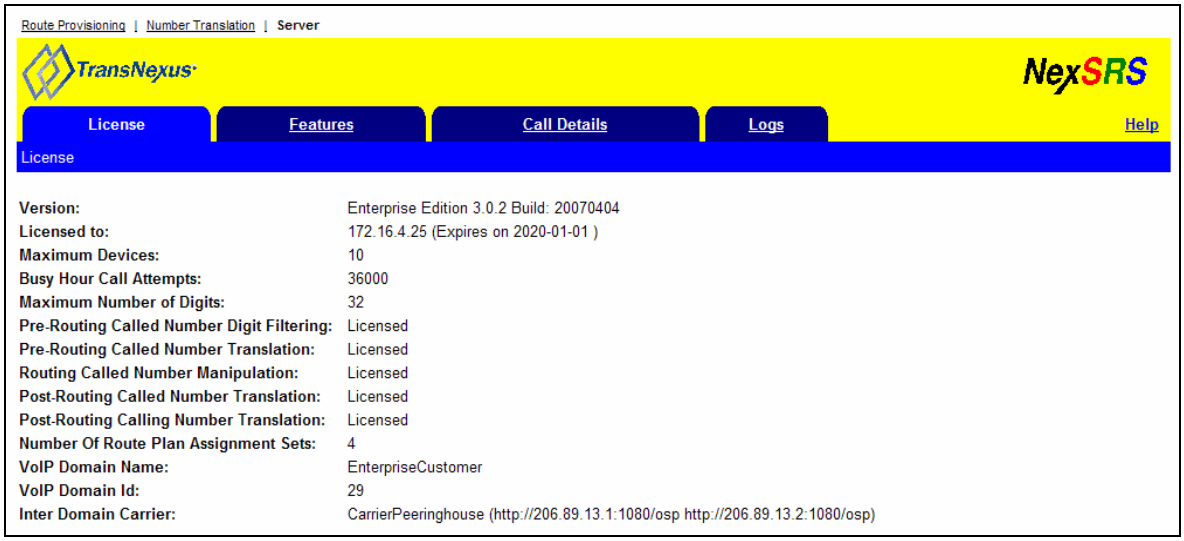
1. The Source VoIP Device on the Enterprise IP network sends an OSP *AuthorizationRequest* to the Enterprise NexSRS server.
2. The Enterprise NexSRS server will first search its own routing table for a device to terminate the call. If no route is found, the Enterprise NexSRS server will automatically send a second OSP *AuthorizationRequest* to the Carrier of its choice. If the first Carrier NexSRS server does not respond, the Enterprise NexSRS server will resend the OSP *AuthorizationRequest* to the second Carrier NexSRS server.
3. The Carrier NexSRS server receives the OSP *AuthorizationRequest* from the Enterprise NexSRS server, finds a route for Source VoIP Device and returns an OSP *AuthorizationResponse* to the Enterprise NexSRS server.
4. The Enterprise NexSRS server repackages the routing information and tokens from the Carrier NexSRS server into its own OSP *AuthorizationResponse* to the Source VoIP Device. The Source VoIP Device cannot detect a difference between internal routes and Extra-Net routes.
5. The Source VoIP Device uses the routing information and tokens provided by the Enterprise NexSRS server to set-up a call to the Destination VoIP Device on the Carrier IP network. The Destination VoIP Device accepts the call set-up because the token has been signed by its Carrier NexSRS server.
6. The VoIP call is set-up directly between the Source and Destination VoIP Devices.

When the call is complete, the Source VoIP Device will send an OSP UsageIndication to the Enterprise NexSRS server and the Destination VoIP Device will send an OSP UsageIndication to the Carrier NexSRS server.

How to configure an Enterprise NexSRS Server

Configuring an Enterprise NexSRS server to communicate to a carrier NexSRS server requires two simple steps. First, install a new license which includes the inter-OSP network details of

your inter-domain IP carrier. Below is an example of a license which enables an Enterprise NexSRS server to query a carrier NexSRS server for routes.



In the license above are three fields which enable Extra-Net routing. These fields are:

VoIP Domain Name: This is the name of the Enterprise NexSRS server operator.

VoIP Domain ID: This is a unique ID assigned to the Enterprise VoIP network.

Inter Domain Carrier: This is the name of Carrier who will provide ExtraNet routes and the IP addresses of Carrier NexSRS servers.

As soon as the new license is installed, the Enterprise NexSRS server will begin querying the Carrier NexSRS servers for ExtraNet routes when termination routes do not exist in the Enterprise NexSRS server. If a route does not exist in the Enterprise NexSRS server, it must be explicitly blocked to avoid ExtraNet routing.

The second step is required only if the Enterprise NexSRS server is behind a Network Address Translation (NAT) device such as a firewall or load balancer. When the Enterprise NexSRS server sends an OSP *AuthorizationRequest* to a Carrier NexSRS server, the Enterprise NexSRS server identifies itself by including it's public IP address in the OSP *AuthorizationRequest* message. If the IP address of the NexSRS server is different from it's public IP address, then the public IP address of the Enterprise NexSRS server must be explicitly configured in the `inter_osp_config.sh` file. The `SERVER_PUBLIC_IP_ADDRESS` parameter in the `inter_osp_config.sh` file must be changed to the public IP address of the Enterprise NexSRS server as seen by external networks. If this parameter is changed to a public IP address, the address must be entered in brackets, for example, [123.123.123.123]. This parameter may also be specified as a fully qualified domain name that can be resolved by any public Domain Name Server, for example, osptestserver.transnexus.com. The `inter_osp_config.sh` file is included in Appendix 4.

Once the Enterprise NexSRS server is configured for ExtraNet routing, calls that can not be routed by the Enterprise NexSRS server will be routed by the Carrier NexSRS server to

termination devices in the Carrier VoIP domain. If calls are not routed by the Carrier NexSRS server, the problem may be due to a configuration error in the Carrier NexSRS server. Please see the ExtraNet log messages (401 Unauthorized) in Chapter 10 for information on how to recognize Carrier NexSRS server configuration errors.

How to Configure a Carrier NexSRS server for ExtraNet Routing

An Enterprise NexSRS server should be configured in the Carrier NexSRS server as a VoIP Device that may not terminate VoIP calls. The Enterprise NexSRS server must be assigned to a Customer that may originate VoIP calls. Since Enterprise OSP to Carrier NexSRS server communication is authorized through the Enterprise NexSRS server license, it does not enroll with the Carrier NexSRS server. Therefore, the Carrier NexSRS server operator must manually set the status of the Enterprise NexSRS server to enrolled by clicking on Has Enrolled check box for the Enterprise NexSRS device on the **Devices** page of the NexSRS user interface.

ENUM Module

The NexSRS Peering Server can also be enabled to query ENUM servers for external routing alternatives. The ENUM lookup feature is provided in a separate module for the NexSRS server. Contact support@transnexus.com for information.

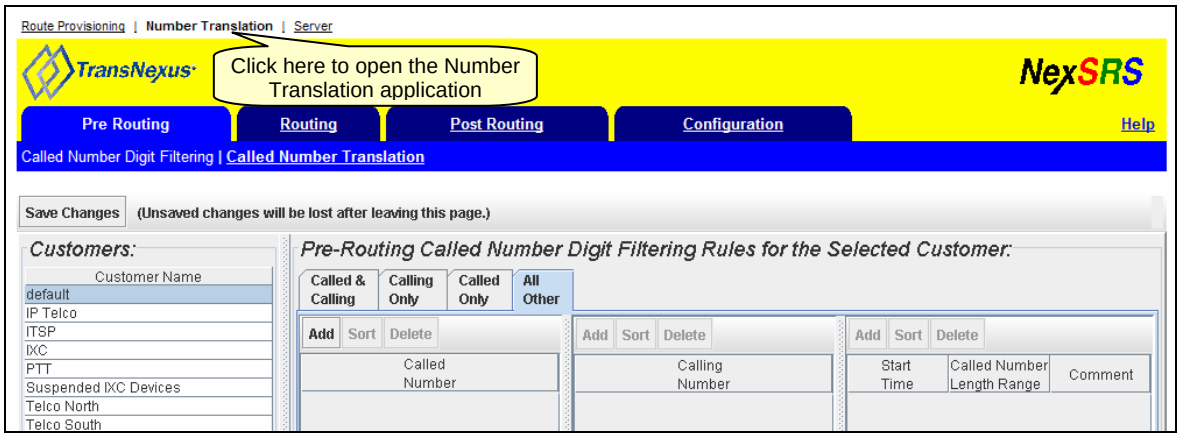
CALEA Module

A high performance CALEA module is available for the NexSRS Peering Server which allows secret monitoring and re-routing of VoIP calls for law enforcement purposes. The CALEA module can re-route calls based on any combination of up to two million calling numbers, called numbers, source IP addresses or destination IP addresses. Contact support@transnexus.com for more information.

Chapter 6

Number Translation

This chapter describes the NexSRS Number Translation application. To open the Number Translation application, click on the link at the top of the NexSRS web interface.



Number Translation features are independent from the Dial Plan and Product features of the Route Provisioning application. However, the features of the two applications are used together to provide the NexSRS operator with a wide range of routing and call treatment flexibility. Number Translations are divided into three categories:

- **Pre-Routing** number translation rules take effect before the routing algorithm determines the call destination.
- **Routing** manipulation rules takes effect during the routing algorithm.
- **Post-Routing** number translation rules take effect after the routing algorithm determines the call destinations.

Like Dial Plans, Number Translation rules can be applied based on:

1. The combination of the Original Called Number and Calling Number.
2. The Original Calling Number only
3. The Called Number only
4. All Other – a global translation rule for all cases not specified in rules 1, 2, or 3.

The translated number can be any combination of the digits 0-9, and the alpha characters hash (#), asterisk (*) and comma (,). In addition, different number translation rules can be

defined based on time of day and day of week. The maximum length for calling, called and translated numbers is 32 digits.

Unique number translation rules are specified for each Customer. In addition, number translation rules for the default Customer (default spelled in lower case) apply to all Customers when Customer specific translation rules do not exist. The Number Translation rule selection algorithm is very similar to the Dial Plan selection algorithm. Please see Appendix 6 for a detail description.

The following table describes the sequence of events that occur in the NexSRS server when a routing query is received. The Number Translation events are highlighted in yellow. In Tech Prefix and Out Tech Prefix events are also included in this table. In Tech Prefix and Out Tech Prefix translations are device properties and not features of the Number Translation application. However, they are included in this table for completeness.

NexSRS Number Translation and Routing Events

	Event	**Description**
1	Receive Routing Query from source device	The called number in the route query will be the called number reported in the CDR from the source device.
2	Remove In Tech Prefix from Called Number	In Tech Prefix must be configured for the source device and configured to be stripped as a property of the source device.
3	Pre-routing called number filtering	Block the call if the called number is too long or too short to be a valid telephone number.
4	Pre-routing called number translation	Translate the called number.
5	Routing called number manipulation	Translate the called number, but only for the routing algorithm process.
6	Select Dial Plan for routing	Use the translated called number for Dial Plan selection and call routing.
7	Undo routing called number manipulation	Reverse the routing called number manipulation (event 5).
8	Post-routing called number translation	Translate the called number.
9	Post-routing calling number	Translate the calling number
10	Add Out-Tech prefix	If an Out Tech Prefix is configured for the destination device, add the Out Tech Prefix to the called number.
11	Return Routing Response to source device	The called number returned to the source device will be the called number reported in the CDR from the destination device.

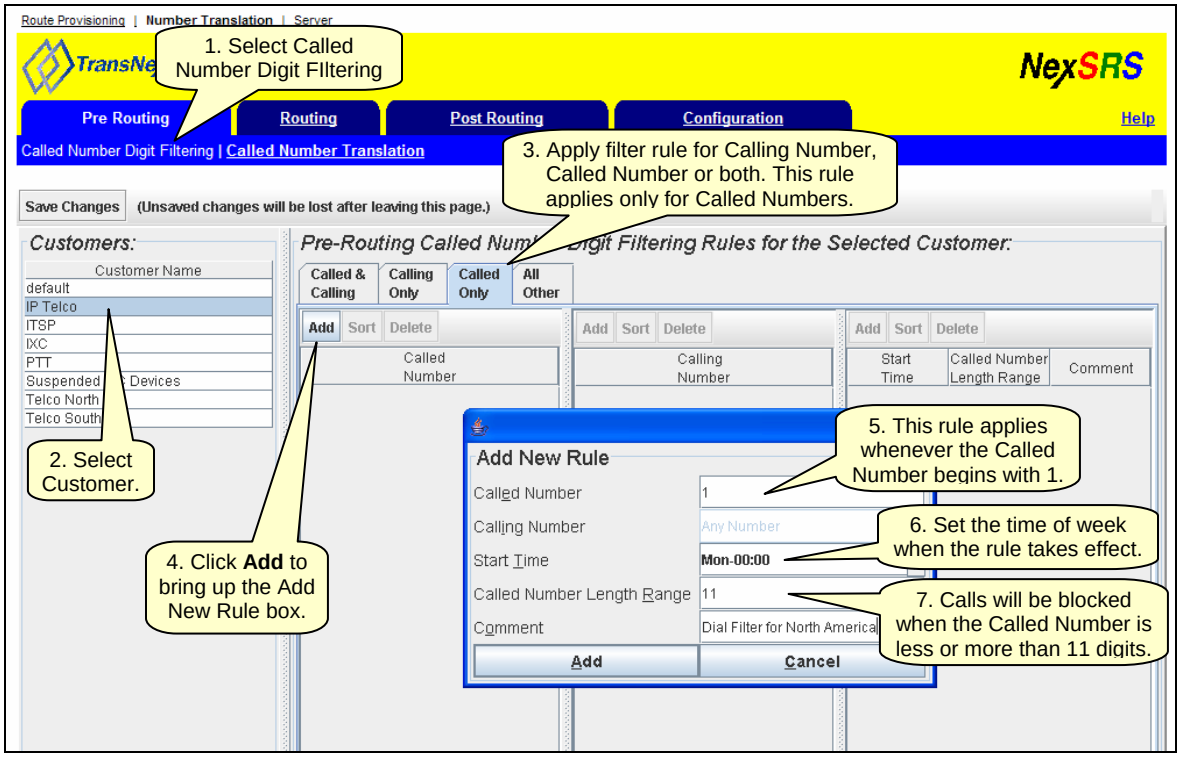
Pre-Routing Rules

The **Pre-Routing** tab of the NexSRS server provides two sets of call treatment rules that are applied before the routing algorithm. These rules are Called Number Digit Filtering and Called Number Translation. In the previous table, these are events 3 and 4.

Called Number Digit Filtering

Called Number Digit Filtering is a filter rule which can be configured to block mis-dialed calls. For example, if all calls to country code 1 (North America) telephone numbers should have eleven digits, then a filter can be created that will automatically block mis-dialed calls, beginning with 1, that have fewer or more than 11 digits. This feature conserves network resources by eliminating mis-dailed calls before call setup.

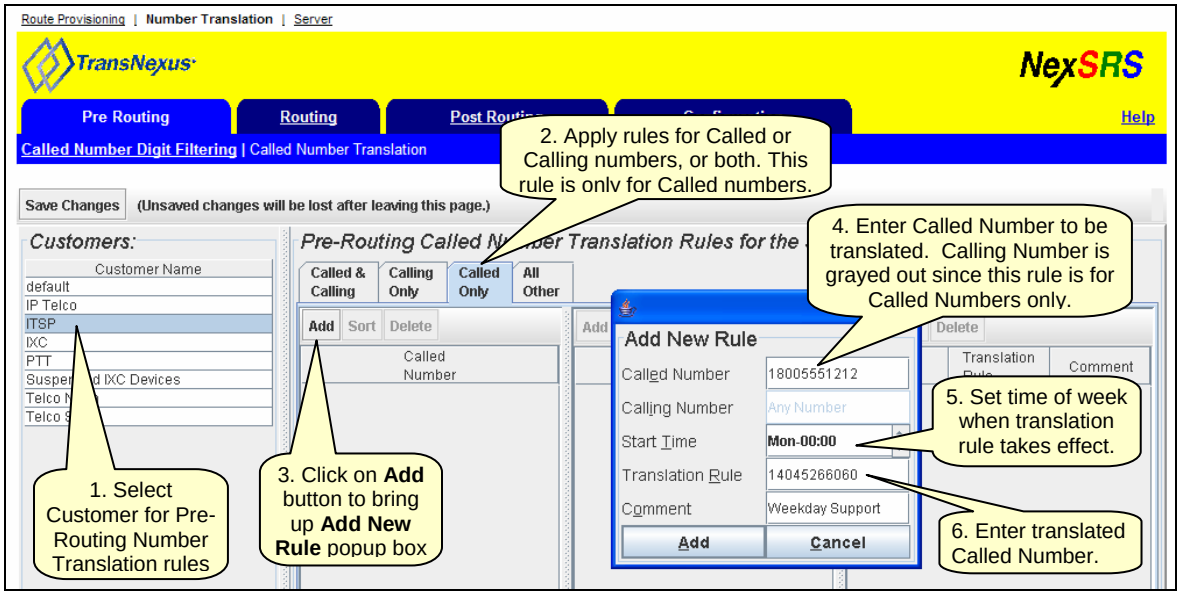
The screen shot below illustrates how to create this filtering rule. The first step is to select the Customer for which the rule will apply. In the example below, the IP Telco Customer has been selected. Next, the **Called Only** sub-tab is selected since this rule only applies to the called number. The **Add** button is selected to bring up the **Add New Rule** box. The number 1 is entered in the Called Number field – this rule will apply to any called number beginning with 1. The Calling Number field is grayed out since this rule is based only on the called number. The Start Time field specifies what time and day of week the rule will take affect. This rule is configured to start at 12:00 am midnight on Monday. The Called Number Length Range specifies how many digits the called number should have. The Called Number Length Range can be specified as a range of integers from minimum to maximum or as a single integer. If the called number has less or more digits than the specified range, the call will be blocked.


1. Select Called Number Digit FIltering
3. Apply filter rule for Calling Number, Called Number or both. This rule applies only for Called Numbers.
2. Select Customer.
4. Click Add to bring up the Add New Rule box.
5. This rule applies whenever the Called Number begins with 1.
6. Set the time of week when the rule takes effect.
7. Calls will be blocked when the Called Number is less or more than 11 digits.
Add New Rule
Called Number
Calling Number
Start Time
Called Number Length Range
Comment
Add
Cancel

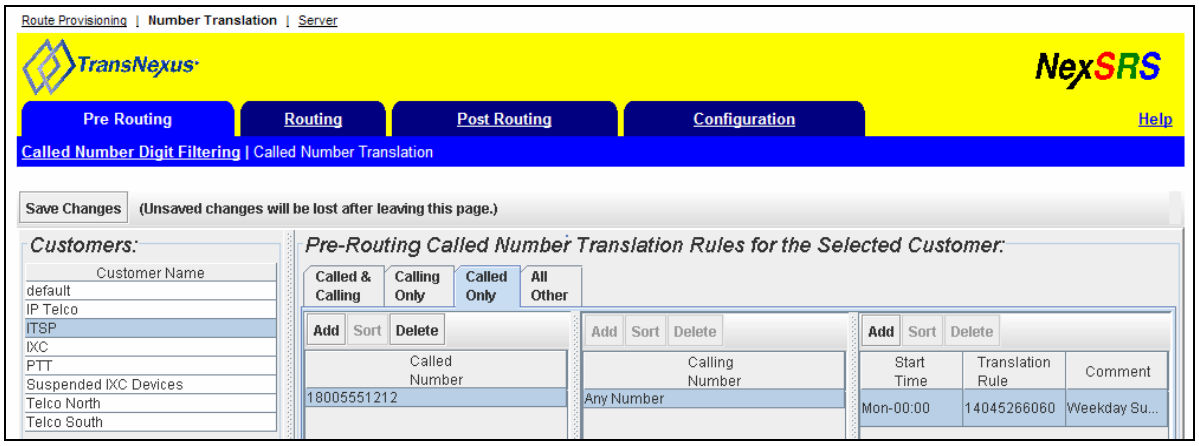
Called Number Translation

Called Number Translation is the second pre-routing rule. The NexSRS server can be configured to replace the called number with a translated called number before the call destination has been determined from the routing table. Follow the steps below to add a Number Translation Rule

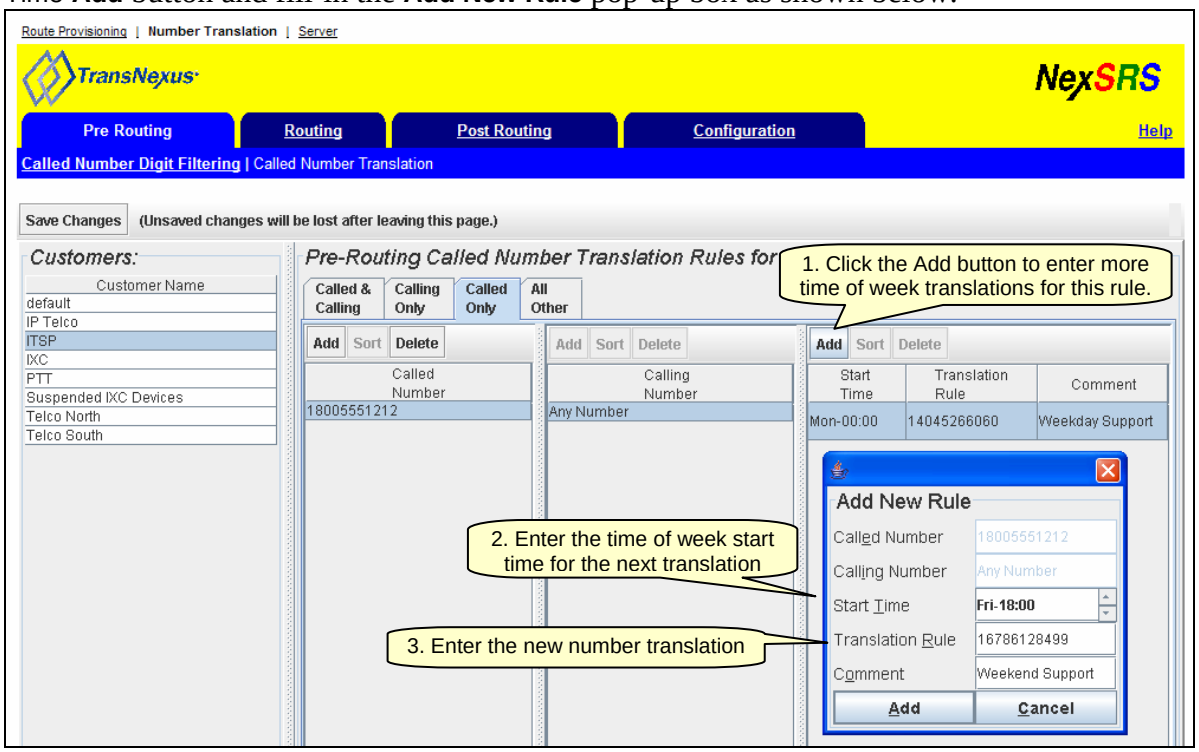
1. Select a Customer. In the following screenshot, the ITSP Customer is selected.
2. Select a Translation Rule. Click on one of the Translation Rules tabs:
   a. **Called & Calling** – Translation based on both called and calling number.
   b. **Calling Only** – Translation based on calling number only.
   c. **Called Only** – Translation based on called number only.
   d. **All Other** – Translation based on any calling or called number
3. Add a New Rule. Select the **Add** button to bring up the **Add New Rule** box. Fill-in the fields in the **Add New Rule** box as shown below to create the number translation rule.



Below is a screenshot of the provisioned Pre-Routing Called Number Translation Rule.

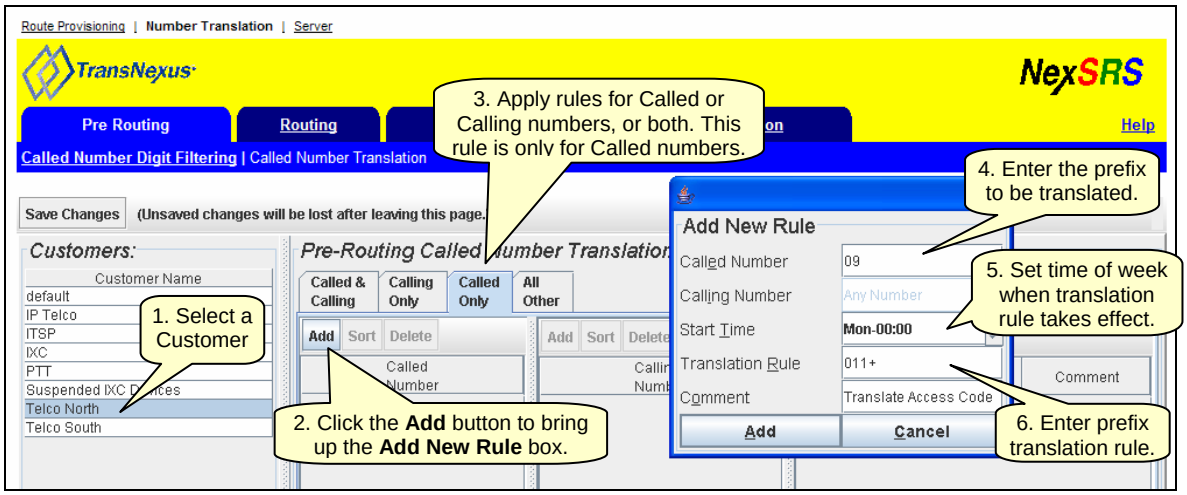


To expand the number translation rule for different times of the week, click on the Start Time **Add** button and fill-in the **Add New Rule** pop-up box as shown below.



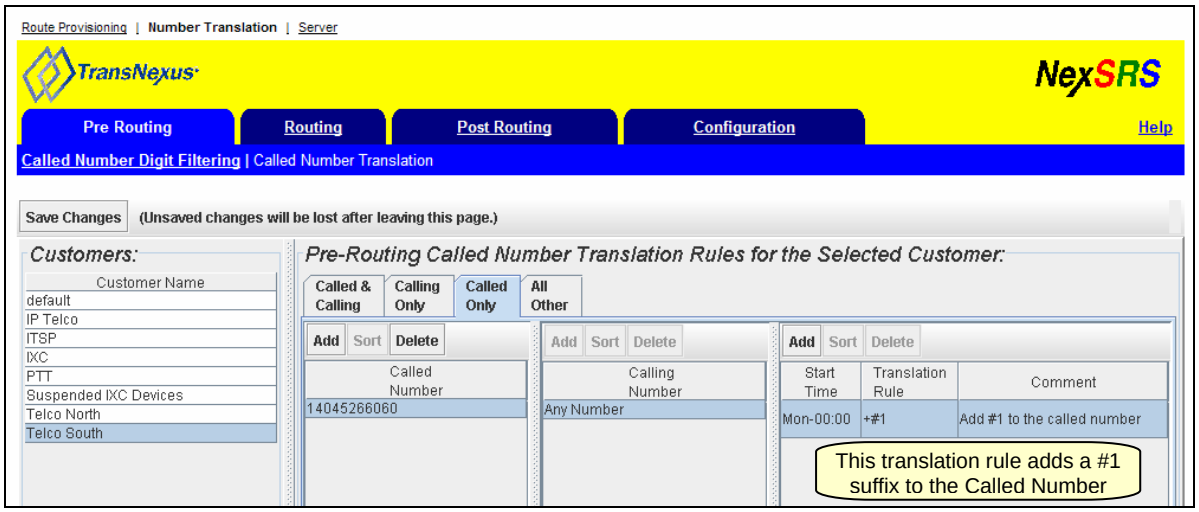
Number Prefix Translation

Number prefix translation rules are provisioned by adding a + to the end Translated Number prefix. The + sign indicates that only the prefix should be translated and that the remaining digits in the called or calling number should not be changed. The following example illustrates called number prefix translation rule that replaces the called number prefix 09 with 011. For example, using the translation rule below, the called number **09**4045266060 would be translated to **011**4045266060.



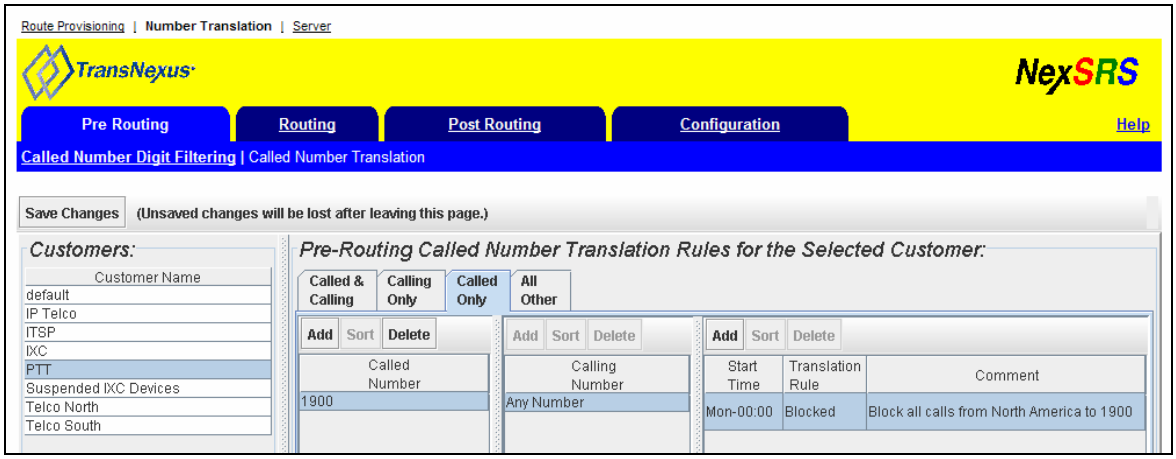
Called Number Suffix Addition

A suffix may be added to called numbers by adding characters after the + sign. For example, if the translated rule is +#1 then called numbers will be translated to have #1 added to the end of the original called number. For example, the rule below will translate called numbers, from VoIP devices in the Telco South Customer, from 14045266060 to 14045266060#1.



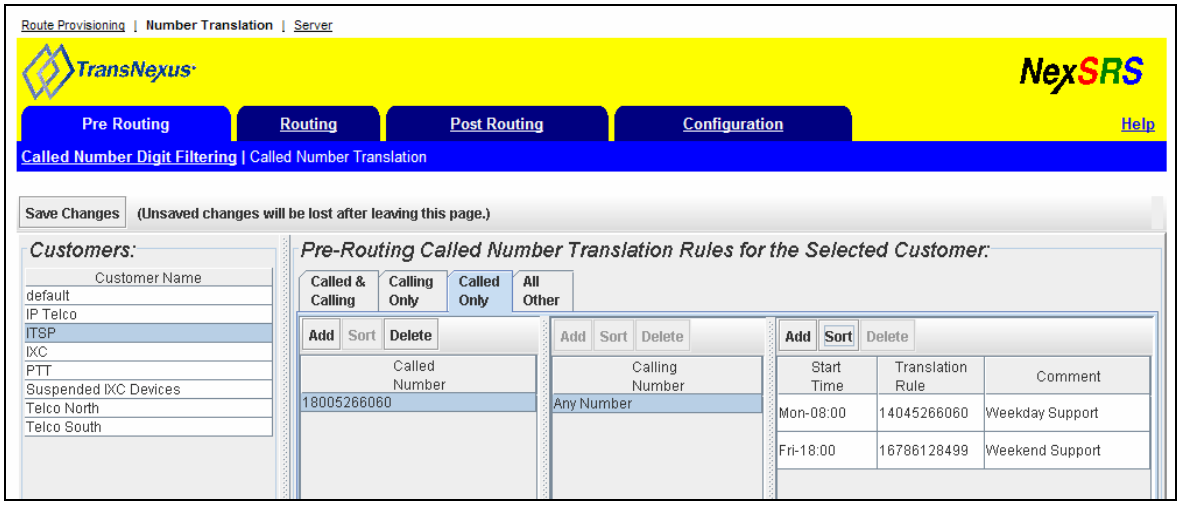
Blocking Calls with Number Translation

Number Translation may be used to block calls when the Translated Called Number is set to 'Blocked'. In the following example, all calls from the PTT Customer with a calling number beginning with 1 and called number beginning with 1900 will be blocked.



Number Translation for Call Centers

This rule is very useful for many services such as translation of 800 number services. Below is an example of Pre-Routing Called Number Translation Rules configured for an 800 customer support number. Starting at 08:00 hours on Monday morning the customer support number is translated to 14045266060 for weekday customer support. Starting at 18:00 on Friday evening, the 800 customer support number is translated 16786128499 for weekend customer support.



Routing Called Number Manipulation Rules

The Called Number Manipulation link under the **Routing** tab is similar to the pre-routing called number translation feature, except that the called number change is applied temporarily during the routing algorithm. After the call route lookup is complete, the called number is restored to what it was before the manipulation. With number translation, the change to the called number is permanent.

Called number manipulation is very useful for addressing special routing situations which are based on the calling number. For example, termination rates for inter-state and intra-state calls are often different. Therefore, calls to a given telephone number will be routed differently depending on whether the call was originated inside or outside the state of the called party. This requires two Products – an int**er**-state routing Product and an int**ra**-state routing Product. But, how to determine when to use which Product?

One solution is to prefix all the breakouts in the intra-state Product with a flag, such as ##. Then use called number manipulation to analyze the calling and called number to determine if the call is an inter or intra-state call. If the call is intra-state, use called number manipulation to add a ## prefix to the called number during the routing process. In the Customer's Dial Plan, configure all calls to any called number beginning with ## to be assigned to the int**ra**-state Product. All other Dial Plan breakouts, with no prefix, should be assigned to the int**er**-state Product. After the call route is selected using the intra-state Product, the ## prefix is removed automatically by the called number manipulation feature.

Post-Routing Rules

Post-routing rules include Calling Number and Called number translation. These features are identical to pre-routing number translation, except that the number translation occurs after the termination devices have been found in the routing table. For detailed information on how to configure number translation rules, please see the pre-routing number translation section.

Chapter 7

NexSRS Server Operation

Operation of the NexSRS server requires minimal effort to operate after configuration. The primary operating tasks are adding and deleting Customers and devices, managing routes and archiving or deleting CDR files. This chapter describes operating scripts, NexSRS User Interface pages and utility scripts used for managing the NexSRS server operation.

Operating Scripts

Starting the NexSRS Server

The NexSRS server must always be started using the `start_osp_server.sh` script. This shell script is required to initialize parameters configured for the NexSRS server. To run this script the operator must own the nexus_server application and be located in the $NexSRS_HOME directory. Type the following command to start the NexSRS server: `./start_osp_server.sh`

To check if the server has started, type this command: `ps -ef|grep nexus`

If the NexSRS server is running you will see both the `nexus_server` and `nexus_watch_dog` applications are alive. Below is an example of what to expect if the NexSRS server is operating:

```
>ps -ef|grep nexus
    root 26612     1  0   Nov 19 ?         0:00 ./nexus_watch_dog
    root 26617 26612  0   Nov 19 ?       652:23 nexus_server
```

Stopping the NexSRS Server

Use the `stop_osp_server.sh` script to stop the NexSRS server. This script automates the process of shutting down the NexSRS server and making certain the NexSRS server process has exited. The user of this script must own the NexSRS application.

To run the script type: `./stop_osp_server.sh` from the $NexSRS_HOME directory.

The script will ask the NexSRS server to stop and print `OSP Server has stopped` when the NexSRS server is shutdown. If the NexSRS server is not running when the script is run, the user will see the message `OSP Server is not running`.

Automatic start-up and shutdown

The `auto_start_stop_osp_server.sh` script automatically starts the NexSRS server when the UNIX server is powered up and automatically stops the NexSRS server when the UNIX server is powered down. This script is located in the `$NexSRS_HOME` directory.

How to configure this script:

1. Edit the script and set the `PATH_TO_OSP_SERVER_DIRECTORY` variable in the script to the directory of the NexSRS server. When using the partition and softlink configuration recommended by TransNexus, this variable should be set to `/OSP/ospnexus`.

2. To configure automatic start-up, create a softlink from the `/etc/rc3.d/SnnOSPServer` directory to `auto_start_stop_osp_server.sh` where "nn" is a two digit integer value not used by other S scripts in the `/etc/rc3.d` directory. An example softlink is shown below. This command must be entered by a user with root privileges.

```
ln -s /OSP/ospnexus/auto_start_stop_osp_server.sh /etc/rc3.d/S51OSPServer
```

3. To configure automatic shutdown, create a softlink from the `/etc/rc0.d/KnnOSPServer` directory to `auto_start_stop_osp_server.sh` where "nn" is a two digit integer value not used by other K scripts in the `/etc/rc0.d` directory. An example softlink is shown below. This command must be entered by a user with root privileges.

```
ln -s /OSP/ospnexus/auto_start_stop_osp_server.sh /etc/rc0.d/K51OSPServer
```

Log Messages

Log messages are the best way to observe the past and current status of the NexSRS server. If problems arise with the NexSRS server, the log messages are the first resource the operator should use for determining the cause of a problem. The log messages are descriptive and provide useful information for identifying problems. Please see Chapter 10 for a listing of log messages and remedies for errors identified by log messages.

Current log messages can be viewed on the Logs page under the **Server** tab on the NexSRS server user interface. This page displays the last 120 log messages recorded in the nexus.log file. An example screen shot showing the logs page during routine operation is shown on the next page. A "heartbeat" log message is written each minute indicating how many call attempts have occurred during the previous minute and hour. Below is an example of a log message showing the call attempts for the previous minute and hour.

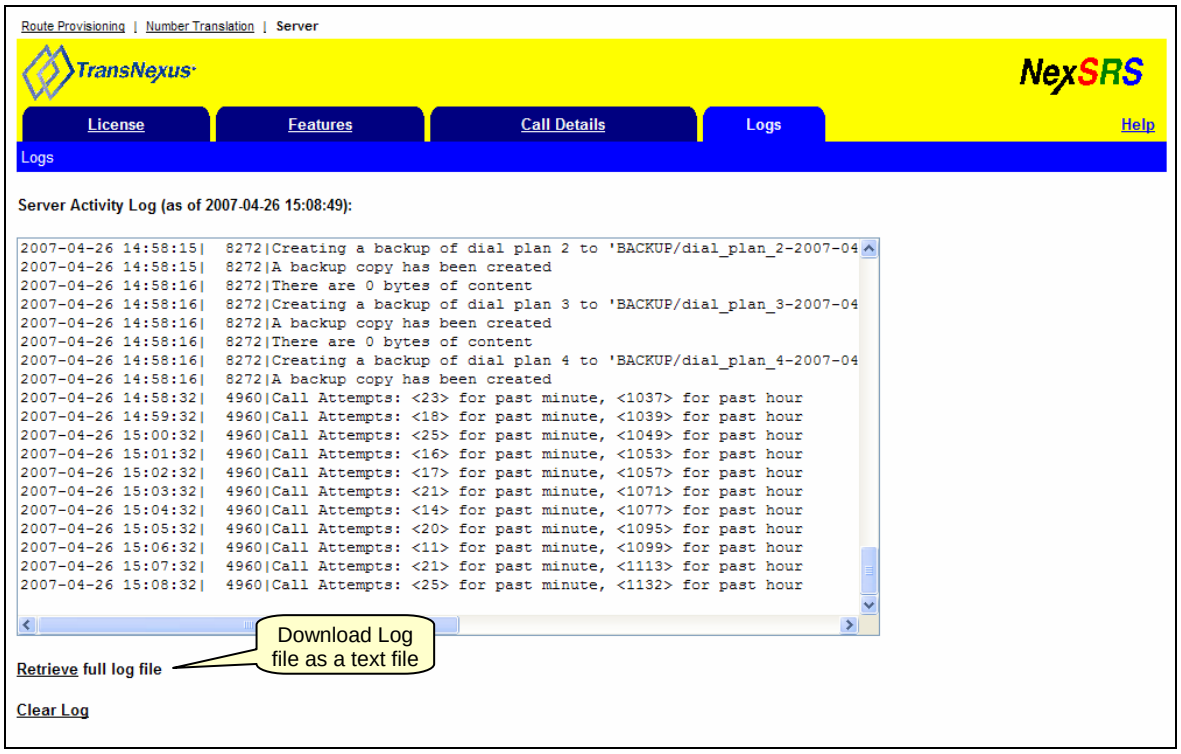
```
2001-11-29 15:53:23|   4|Call Attempts: <52> for past minute, <637> for past hour
```

Current log messages may also be viewed from the $NexSRS_HOME directory of the NexSRS server by typing the following UNIX command.

```
tail -50f xitami/osp_ui/log/nexus.log
```

Current log messages are stored in the `nexus.log` file which may be downloaded by clicking on the Retrieve full log file link of the user interface. The `nexus.log` file may also be accessed directly in the `$NexSRS_HOME/xitami/osp_ui/log` directory of the NexSRS server. Past log files are also stored in this directory using the file names of nexus.logYYYYMMDD-HH:MM

where the date/time stamp indicates when the log was closed and written to file. The Clear Log link clears the Logs page and the nexus.log file of all log messages. Access to the Clear Log link should be limited to the NexSRS administrator (see Chapter 9 on User Administration).


Route Provisioning | Number Translation | Server
TransNexus
NexSRS
License
Features
Call Details
Logs
Help
Logs
Server Activity Log (as of 2007-04-26 15:08:49):
2007-04-26 14:58:15| 8272|Creating a backup of dial plan 2 to 'BACKUP/dial_plan_2-2007-04
2007-04-26 14:58:15| 8272|A backup copy has been created
2007-04-26 14:58:16| 8272|There are 0 bytes of content
2007-04-26 14:58:16| 8272|Creating a backup of dial plan 3 to 'BACKUP/dial_plan_3-2007-04
2007-04-26 14:58:16| 8272|A backup copy has been created
2007-04-26 14:58:16| 8272|There are 0 bytes of content
2007-04-26 14:58:16| 8272|Creating a backup of dial plan 4 to 'BACKUP/dial_plan_4-2007-04
2007-04-26 14:58:16| 8272|A backup copy has been created
2007-04-26 14:58:32| 4960|Call Attempts: <23> for past minute, <1037> for past hour
2007-04-26 14:59:32| 4960|Call Attempts: <18> for past minute, <1039> for past hour
2007-04-26 15:00:32| 4960|Call Attempts: <25> for past minute, <1049> for past hour
2007-04-26 15:01:32| 4960|Call Attempts: <16> for past minute, <1053> for past hour
2007-04-26 15:02:32| 4960|Call Attempts: <17> for past minute, <1057> for past hour
2007-04-26 15:03:32| 4960|Call Attempts: <21> for past minute, <1071> for past hour
2007-04-26 15:04:32| 4960|Call Attempts: <14> for past minute, <1077> for past hour
2007-04-26 15:05:32| 4960|Call Attempts: <20> for past minute, <1095> for past hour
2007-04-26 15:06:32| 4960|Call Attempts: <11> for past minute, <1099> for past hour
2007-04-26 15:07:32| 4960|Call Attempts: <21> for past minute, <1113> for past hour
2007-04-26 15:08:32| 4960|Call Attempts: <25> for past minute, <1132> for past hour
Download Log file as a text file
Retrieve full log file
Clear Log

The nexus.log file will grow continuously and after several months the log file could become very large. Very large nexus.log files are difficult to download and analyze. Therefore, TransNexus recommends that the current nexus.log file be archived once each month to allow the start of a new nexus.log file. Please see the Utilities section for a script that automates this process.

Backing-up Configuration Files

The NexSRS server has five routing configuration files and five number translation files.

Routing configuration files used by the Route Provisioning application:

1. nexus.cfg (NexSRS server routing table)
2. dial_plan_1.cfg
3. dial_plan_2.cfg
4. dial_plan_3.cfg
5. dial_plan_4.cfg

Number Translation configuration files:

1. pre_routing_called_number_digit_filtering.cfg
2. pre_routing_called_number_translation.cfg
3. routing_called_number_manipulation.cfg

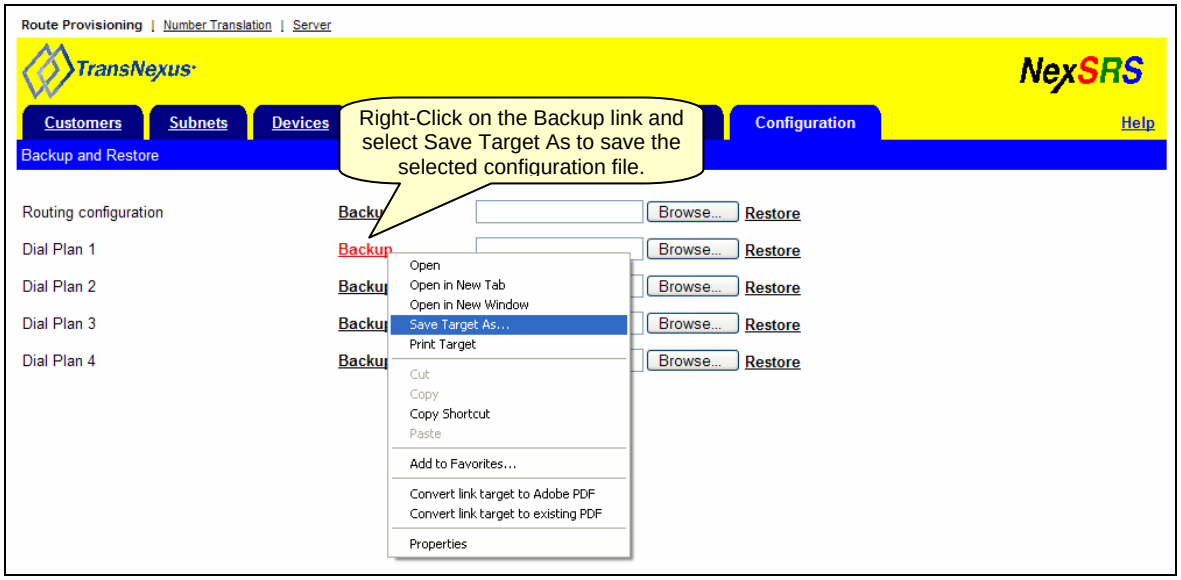
4. post_routing_calling_number_translation.cfg
5. post_routing_called_number_translation.cfg

These files are automatically backed up by the NexSRS server at the following times:

1. Right after start-up
2. Just before shutdown
3. Before the commercial license is updated
4. Before a configuration file is restored from the user interface.
5. Before a configuration file is replaced using NexOSS config-push.

Back-up copies of the files are stored in the `$NexSRS_HOME/BACKUP` directory of the NexSRS server and are identified with a time/date stamp and `.bu` extension (i.e. `nexus_cfg_2001-11-28.bu`). These files are compressed by the `compress_backup_file.sh` script which adds a .gz extenstion to the compressed file's name. Back-up copies of configuration files should be stored in a second location for redundancy.

Back-up copies of the current configuration files can be easily stored to a local PC by clicking the right-hand button of the mouse on the Backup link on the **Configuration** page of the Route Provisioning and Number Translation applications. An example screenshot is shown below. By clicking on the Backup link, the operator will be prompted to download and save the configuration file on the user's local PC as shown below. The Restore link on this page is used to restore previously backed-up configuration files to the NexSRS server.



Normal operation of the NexOSS routing_table_push application will create a continuously growing number of configuration backup files. Deletion of old configuration back-up files is a required maintenance process. Please see the next section for details on the operating script which automates this process.

Utilities

In the $NexSRS_HOME/utils directory are six shell scripts which may be scheduled with a cron job to automate routine management of the NexSRS server. The utility scripts must be run from the $NexSRS_HOME/utils directory by a user ospadmin. See Appendix 7 for the details of each utility script.

compress_archived_cdr_files.sh

This script uses the gzip utility to compress all files with a .cdr extension in the `$NexSRS_HOME/xitami/osp_ui/cdrs/ARCHIVE` directory.

delete_old_cdr_files.sh

This script automates the process of deleting CDR files from the `$NexSRS_HOME/xitami/osp_ui/cdrs/ARCHIVE` directory that have been unchanged for more days than the number of days defined by the `DELETE_FILES_OLDER_THAN_DAYS` variable in the script. The default value for this variable is 45 days and may be changed by editing the script. For example, if this script is run, with `DELETE_FILES_OLDER_THAN_DAYS=45,` all *.cdr files, compressed and uncompressed, that have not been changed, compressed, renamed, etc. for more than 45 days will be deleted.

cut_nexus_log.sh

When this file is run, the current nexus.log file is renamed and then gzipped and saved in the `$NexSRS_HOME/xitami/osp_ui/log` directory. The file name for the archived log file is nexus.log-YYYYMMDD-HH:MM.gz, where YYYYMMDD-HH:MM stands for the year, month, day, hour and minute when the log file was archived up. For example, `nexus.log-20040330-23:15.gz`. When the log file is archived a new log file will automatically be written in the `/xitiami/osp_ui/log` directory.

delete_archived_nexus_log_files.sh

This script automates the process of deleting all archived nexus.log files. This script deletes all files from the `$NexSRS_HOME/xitami/osp/ui` directory that have a .gz extension and have been unchanged for more days than the number of days defined by the DELETE_FILES_OLDER_THAN_DAYS variable. The default value for this variable is 45 days. For example, if DELETE_FILES_OLDER_THAN_DAYS=45 all .gz files, that have not been changed, compressed, renamed, etc. for more than 45 days will be deleted when this script is run.

compress_backup_files.sh

This script uses the gzip utility to compress all files with a .bu extenstion in the `$NexSRS_HOME?BACKUP` directory. This script must be run by user ospadmin.

delete_old_backup_files.sh

This script automates the process of deleting all files from the `$NexSRS_HOME/BACKUP` directory that have a .bu extension and have been unchanged for more days than the number of days defined by the DELETE_FILES_OLDER_THAN_DAYS variable. The default value for this variable is 45 days. For example, if this script is run with DELETE_FILES_OLDER_THAN_DAYS=45 all .bu files, compressed and

uncompressed, that have not been changed, compressed, renamed, etc. for more than 45 days will be deleted.

IsFileOldEnough.pl

The IsFileOldEnough.pl file is a helper script which is used by the utility scripts. This file must reside in the $NexSRS_HOME/utils directory.

How to schedule utility scripts with cron

The utility scripts can be run manually from $NexSRS_HOME/utils directory by user ospadmin or they can be scheduled to run automatically by a cron routine.

CRON

cron is a unix utility which executes unix commands according to a schedule defined in a crontab file. A crontab file consists of lines with six fields each. The fields in each line are separated by spaces or tabs. The first five fields are integer patterns which specify the time schedule of the cron job. The sixth field defines the unix command to be executed. The syntax for a line in a crontab file is: `mm hh DD MM w "UNIX command to be executed"`

```
Field   Definition           Acceptable values
  mm     minute               0-59
  hh     hour                 0-23 or * for any hour
  DD     day of the month     1-31 or * for any day
  MM     month of year        1-12 or * for any month
  w      day of the week      0-6 (0=Sunday) or * for any day of week
```

STEP 1.

To run the utility scripts, the variable $NexSRS_HOME must be defined in the .profile of user ospadmin and point to the working directory of the NexSRS server. The entry in the .profile should be:

```
NexSRS_HOME=<path of the NexSRS server>
export NexSRS_HOME
```

TransNexus recommends that the NexSRS server be installed in a separate partition named OSP. The current directory of the NexSRS server should be softlinked to a link named ospnexus. The entry in the .profile based on the TransNexus recommendation is:

```
NexSRS_HOME=/OSP/ospnexus
export NexSRS_HOME
```

STEP 2.

Running the NexSRS utility scripts with a cron routine requires that the user ospadmin has its own crontab schedule. Allow creation of a crontab schedule for user ospadmin by creating the file `/etc/cron.d/cron.allow` as root and then add the usernames root and ospadmin to the cron.allow file. The following lines provide an example of Step 1. (NOTE: to save the cron.allow file using vi, type "ESCAPE :wq!". The ESCAPE character does not appear in the example below.)

```
cd /etc/cron.d
vi cron.allow
"cron.allow" [New file]
```

```
root
ospadmin
:wq!
"cron.allow" [New file] 2 lines, 14 characters
```

STEP 3.
The next step is to create a crontab file named ospadmin in the crontabs directory in the following path: `/var/spool/cron/crontabs`. This file will contain lines which schedule the execution of the NexSRS utility scripts. (The [`. ./.profile`] must be included with each script so the environmental setting of ospadmin is used to run the script.)

The lines below provide an example of how to move to the crontabs directory and create a new crontab file named ospadmin. In the example below, the utilities are scheduled to run at 00:00 at the beginning of each month, except for the compress_archived_cdr_files script which is scheduled to run at 00:00 everyday. (NOTE: to save the ospadmin file using vi, type "ESCAPE :wq!". The ESCAPE character does not appear in the example below.)

```
vi ospadmin
0 0 * * * . ./.profile ; $NexSRS_HOME/utils/compress_archived_cdr_files.sh
0 0 1 * * . ./.profile ; $NexSRS_HOME/utils/cut_nexus_log.sh
0 0 1 * * . ./.profile ; $NexSRS_HOME/utils/delete_old_backup_files.sh
0 0 1 * * . ./.profile ; $NexSRS_HOME/utils/delete_old_cdr_files.sh
0 0 1 * * . ./.profile ; $NexSRS_HOME/utils/delete_archived_nexus_log_files.sh
:wq!
"ospadmin" 5 lines, 281 characters
```

STEP 4.
After the ospadmin crontab file is created, the next step is to assigned ospadmin as the owner of the crontab file. The example below shows how to perform Step 4.

```
chown ospadmin:osp ospadmin
```

STEP 5.
Perform a listing of the crontabs directory to verify that a crontab file has been added with user ospadmin as the owner of the file. An example of this step and the expected results are shown below.

```
ls -ltr
total 12
-r--r--r--   1 root     root         750 Feb 10 13:59 lp
-rw-r--r--   1 root     sys          190 Feb 10 13:59 adm
-rw-r--r--   1 root     sys          308 Feb 10 14:27 sys
-r--------   1 root     uucp         404 Feb 10 14:29 uucp
-rw-r--r--   1 root     sys          579 Feb 24 17:09 root
-rw-r--r--   1 ospadmin osp          206 Apr 15 19:11 ospadmin
```

STEP 6.
For the new crontab schedule to take affect, the cron daemom must be stopped and restarted as shown in the example command lines below.

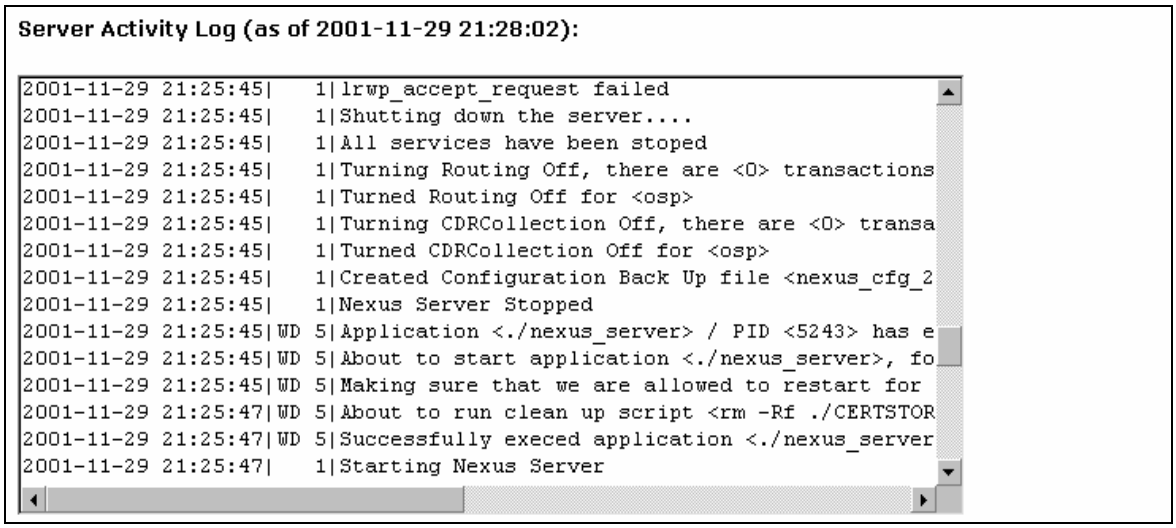
```
ps -ef|grep cron
  root   867     1  0 19:50:01 ?        0:00 /usr/sbin/cron
/etc/init.d/cron stop
```

```
/etc/init.d/cron start

ps -ef|grep cron
   root   453     1  0 19:50:37 ?        0:00 /usr/sbin/cron
```

Watch Dog

A companion application to the NexSRS server is the Watch Dog program. This program constantly monitors the NexSRS server and Xitami web server applications. If either of these applications stop running, they will be immediately restarted by the Watch Dog program. Below is an example of the log messages the NexSRS server operator will see if the Watch Dog detects that the NexSRS server has shut down and is restarted by the Watch Dog program. Messages logged by the Watch Dog program have WD preceding the process thread ID.


Server Activity Log (as of 2001-11-29 21:28:02):
2001-11-29 21:25:45| 1|lrwp_accept_request failed
2001-11-29 21:25:45| 1|Shutting down the server....
2001-11-29 21:25:45| 1|All services have been stoped
2001-11-29 21:25:45| 1|Turning Routing Off, there are <0> transactions
2001-11-29 21:25:45| 1|Turned Routing Off for <osp>
2001-11-29 21:25:45| 1|Turning CDRCollection Off, there are <0> transa
2001-11-29 21:25:45| 1|Turned CDRCollection Off for <osp>
2001-11-29 21:25:45| 1|Created Configuration Back Up file <nexus_cfg_2
2001-11-29 21:25:45| 1|Nexus Server Stopped
2001-11-29 21:25:45|WD 5|Application <./nexus_server> / PID <5243> has e
2001-11-29 21:25:45|WD 5|About to start application <./nexus_server>, fo
2001-11-29 21:25:45|WD 5|Making sure that we are allowed to restart for
2001-11-29 21:25:47|WD 5|About to run clean up script <rm -Rf ./CERTSTOR
2001-11-29 21:25:47|WD 5|Successfully execed application <./nexus_server
2001-11-29 21:25:47| 1|Starting Nexus Server

Chapter 8

Call Detail Records

Introduction

A single VoIP call will generate multiple raw OSP CDRs. These CDRs are streamed to NexSRS servers in real time and written to a CDR file every five minutes. There are four types of CDRs recorded at the NexSRS Server. They are:

Internal CDRs which are generated by the NexSRS Server. An internal CDR is created when a devices requests a route authorization from the NexSRS Server. An internal CDR is created for each possible destination. For example, if there are three possible destinations for a VoIP call the NexSRS Server will create three internal CDRs – one for the authorization to each destination.

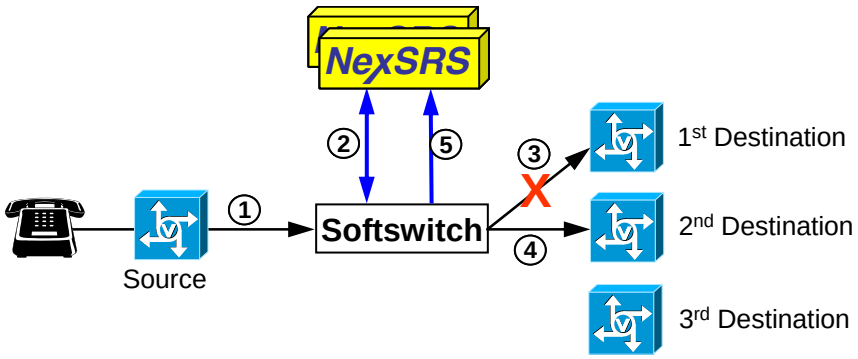
Source CDRs are reported by the VoIP device which makes the routing authorization request to the NexSRS Server. The source device reports a CDRs for each call attempt to a destination. For example, if the source device's call attempt to a destination fails and the source retries the call to a second destination which completes the call, the source device will report two CDRs. One CDR for the failed call attempt and one CDR for the successful call attempt.

Destination CDRs are reported by OSP enabled devices which are the call destination.

Other CDRs are CDRs reported by devices which are neither the call source nor call destination. An OSP CDR reported by a proxy device in the signaling path between the source and destination devices would report an OSP CDR of type "other".

CDRs from a Softswitch

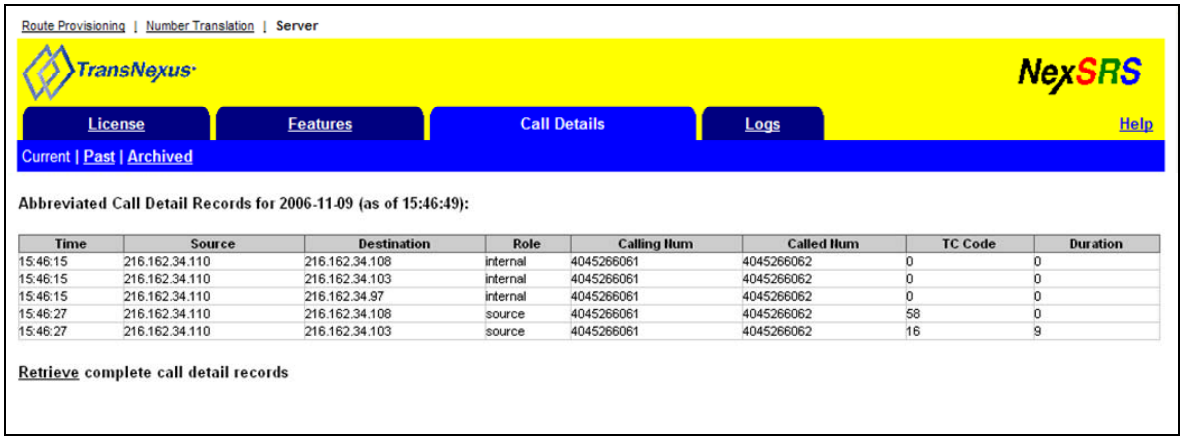
This example explains the CDRs expected when the NexSRS Server is used as a routing and CDR collection server for a softswitch, IP to IP gateway, Back to Back User Agent (B2BUA) or Session Border Controller (SBC). In this example a call is received by the softswitch which sends an OSP peering query the NexSRS Server for routes to destination devices which can complete the call. In this example, there are three possible destinations which can complete the call. The call attempt to the first destination fails, but the softswitch retries the call to the second destination which successfully completes the call. For this call scenario, five CDR are recorded at the NexSRS server.



The following call scenario description explains the CDRs recorded.

1. Calling party originates the call. The source gateway signals the softswitch.
2. The softswitch sends an OSP routing authorization query to the NexSRS Server. The NexSRS Server identifies three possible destination devices which can complete the call. The three destinations are returned in priority order to the softswitch. The NexSRS Server records three internal CDRs – one for each possible destination.
3. The softswitch attempts to complete the call to the first destination, but the call setup fails.
4. The softswitch completes the call to the second destination.
5. When the call is completed, the softswitch reports two CDRs to the NexSRS Server. Once CDR is for the failed call attempt. The second CDR is for the successful call attempt. Both of these CDRs are "source" CDRs since the softswitch is the source device which sent the route query to the NexSRS Server.

A total of five CDRs (three internal and two source CDRs) were generated by this call. Note that two NexSRS Servers are shown in the diagram. NexSRS Servers are stateless route servers designed for distributed load sharing. In this configuration, it is possible to call routing queries (step 2) and CDR messages (step 5) to be sent to different NexSRS Servers.


Route Provisioning | Number Translation | Server

TransNexus · NexSRS

License | Features | Call Details | Logs | Help

Current | Past | Archived

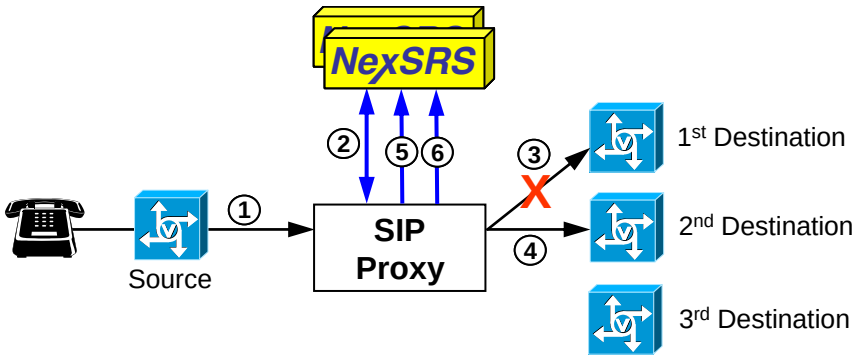
Abbreviated Call Detail Records for 2006-11-09 (as of 15:46:49):

Time	Source	Destination	Role	Calling Num	Called Num	TC Code	Duration
15:46:15	216.162.34.110	216.162.34.108	internal	4045266061	4045266062	0	0
15:46:15	216.162.34.110	216.162.34.103	internal	4045266061	4045266062	0	0
15:46:15	216.162.34.110	216.162.34.97	internal	4045266061	4045266062	0	0
15:46:27	216.162.34.110	216.162.34.108	source	4045266061	4045266062	58	0
15:46:27	216.162.34.110	216.162.34.103	source	4045266061	4045266062	16	9

Retrieve complete call detail records

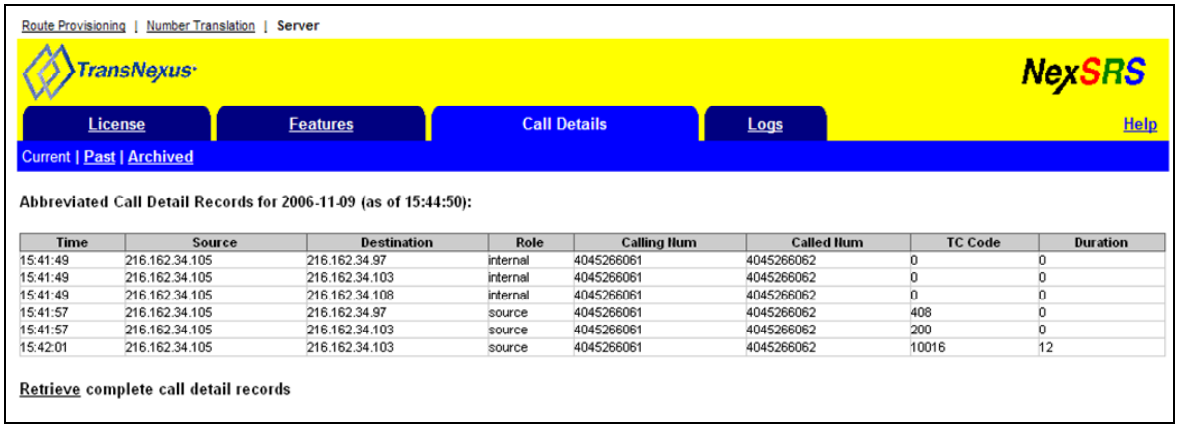
CDRs from a SIP Proxy

The call scenario for a VoIP call routed using a softswitch appears identical to the call scenario using a SIP proxy. However, there is one major difference. A softswitch monitors a VoIP call's state from beginning to end. When the call is completed, the softswitch reports a single CDR for each call attempt. A SIP proxy, however, is a stateless device and does not monitor the progress of a VoIP call. A SIP proxy is aware of the start of a call and the end of the call, but cannot record the start and stop and duration of a VoIP call in a CDR. Instead, a SIP proxy will report a "Start" CDR when a call attempt begins and a "Stop" CDR when a successful call ends. An unsuccessful call attempt will create a "Start" CDR with an error code reported as the Termination Cause Code (TC Code). A successful call setup will generate a "Start" CDR that includes the SIP code of 200 indicating the call setup was successful. At the end of a successful call, the SIP proxy will report a "Stop" CDR with TC Code = 10016. The following call scenario provides an example of CDRs created from a SIP proxy.



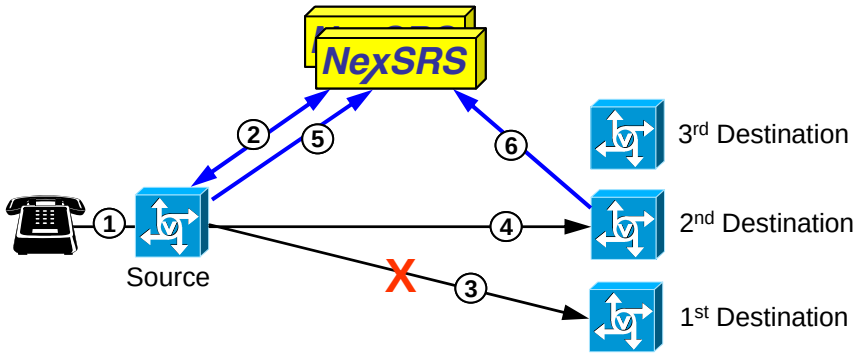
1. Calling party originates the call. The source gateway signals the SIP proxy.
2. The SIP proxy sends an OSP routing authorization query to the NexSRS Server. The NexSRS Server identifies three possible destination devices which can complete the call. The three destinations are returned in priority order to the SIP Proxy. The NexSRS Server records three internal CDRs – one for each possible destination.
3. The SIP proxy attempts to complete the call to the first destination, but the call setup fails. The SIP proxy reports a source "Start" CDR with error code.
4. The SIP proxy completes the call to the second destination.
5. The SIP proxy and reports a source "Start" CDR with TC Code 200.
6. When the call is completed, the SIP proxy reports a source "Stop" CDR with TC Code 10016.

This call scenario generates six CDRs: three internal CDRs, two source "Start" CDRs and one source "Stop" CDR.



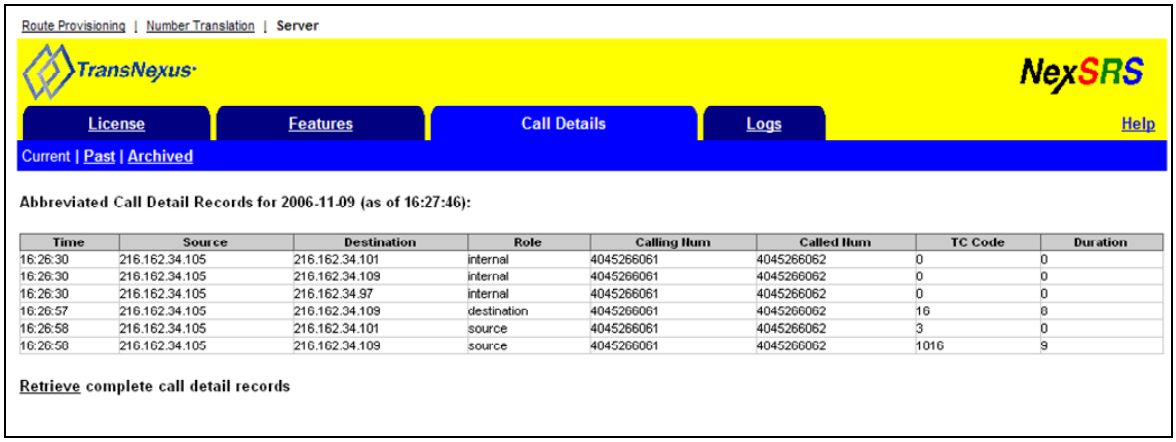
CDRs from Direct Peering

In addition to acting as a route server, the NexSRS Server also enables secure peer to peer routing and CDR collection. This example illustrates the CDRs expected from a direct peer to peer call scenario.



1. The calling party originates the call.
2. The source gateway sends an OSP routing authorization query to the NexSRS Server which responds with a list of three destinations. The NexSRS Server records three internal CDRs.
3. The source attempts to complete the call to the first destination, but the call fails.
4. The source completes the call to the second destination.
5. When the call is completed, the source device sends two source CDRs to the NexSRS Server. One source CDR is for the failed call attempt to the first destination and the second source CDR is for the successful call to the second destination.
6. The destination device which completed the call reports a destination CDR directly to the NexSRS Server.

This call scenario generates six CDRs: three internal CDRs, two source CDRs and one destination CDR.



Route Provisioning | Number Translation | Server

TransNexus · NexSRS

License | Features | Call Details | Logs | Help

Current | Past | Archived

Abbreviated Call Detail Records for 2006-11-09 (as of 16:27:46):

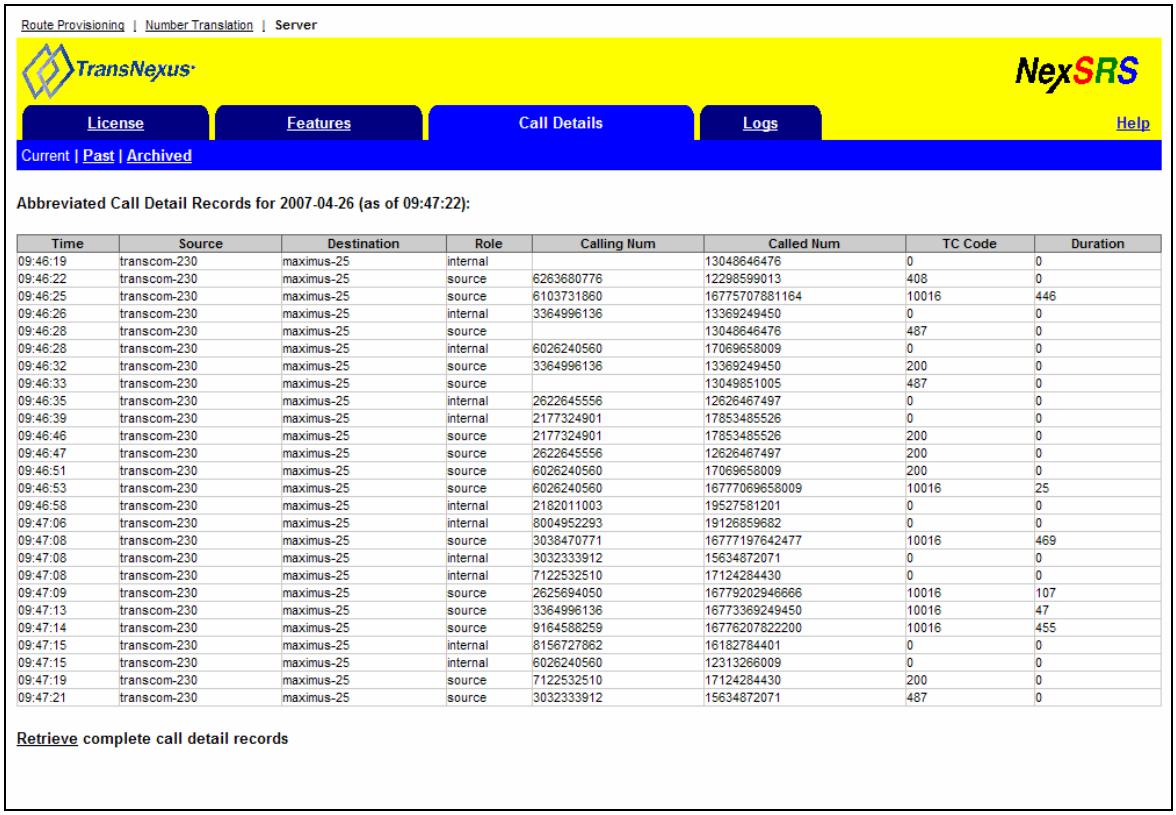
Time	Source	Destination	Role	Calling Num	Called Num	TC Code	Duration
16:26:30	216.162.34.105	216.162.34.101	internal	4045266061	4045266062	0	0
16:26:30	216.162.34.105	216.162.34.109	internal	4045266061	4045266062	0	0
16:26:30	216.162.34.105	216.162.34.97	internal	4045266061	4045266062	0	0
16:26:57	216.162.34.105	216.162.34.109	destination	4045266061	4045266062	16	8
16:26:58	216.162.34.105	216.162.34.101	source	4045266061	4045266062	3	0
16:26:58	216.162.34.105	216.162.34.109	source	4045266061	4045266062	1016	9

Retrieve complete call detail records

Current CDRs

Current call detail records can be accessed from **Call Details** tab of the NexSRS server user interface. The Current link on the **Call Details** Page displays the last 50 call detail records. An example of this page is shown below.

The user may download a file of all of the current call detail records by clicking on the **Retrieve complete call detail records** link at the bottom of the Current CDRs page. A pop-up window is displayed so that the user can enter/select the destination directory to which the tab delimited text file will be saved.



Route Provisioning | Number Translation | Server

TransNexus · NexSRS

License | Features | Call Details | Logs | Help

Current | Past | Archived

Abbreviated Call Detail Records for 2007-04-26 (as of 09:47:22):

Time	Source	Destination	Role	Calling Num	Called Num	TC Code	Duration
09:46:19	transcom-230	maximus-25	internal		13048646476	0	0
09:46:22	transcom-230	maximus-25	source	6263680776	12298599013	408	0
09:46:25	transcom-230	maximus-25	source	6103731860	16775707881164	10016	446
09:46:26	transcom-230	maximus-25	internal	3364996136	13369249450	0	0
09:46:28	transcom-230	maximus-25	source		13048646476	487	0
09:46:28	transcom-230	maximus-25	internal	6026240560	17069658009	0	0
09:46:32	transcom-230	maximus-25	source	3364996136	13369249450	200	0
09:46:33	transcom-230	maximus-25	source		13049851005	487	0
09:46:35	transcom-230	maximus-25	internal	2622645556	12626467497	0	0
09:46:39	transcom-230	maximus-25	internal	2177324901	17853485526	0	0
09:46:46	transcom-230	maximus-25	source	2177324901	17853485526	200	0
09:46:47	transcom-230	maximus-25	source	2622645556	12626467497	200	0
09:46:51	transcom-230	maximus-25	source	6026240560	17069658009	200	0
09:46:53	transcom-230	maximus-25	source	6026240560	16777069658009	10016	25
09:46:58	transcom-230	maximus-25	internal	2182011003	19527581201	0	0
09:47:06	transcom-230	maximus-25	internal	8004952293	19126859682	0	0
09:47:08	transcom-230	maximus-25	source	3038470771	16777197642477	10016	469
09:47:08	transcom-230	maximus-25	internal	3032333912	15634872071	0	0
09:47:08	transcom-230	maximus-25	internal	7122532510	17124284430	0	0
09:47:09	transcom-230	maximus-25	source	2625694050	16779202946666	10016	107
09:47:13	transcom-230	maximus-25	source	3364996136	16773369249450	10016	47
09:47:14	transcom-230	maximus-25	source	9164588259	16776207822200	10016	455
09:47:15	transcom-230	maximus-25	internal	8156727862	16182784401	0	0
09:47:15	transcom-230	maximus-25	internal	6026240560	12313266009	0	0
09:47:19	transcom-230	maximus-25	source	7122532510	17124284430	200	0
09:47:21	transcom-230	maximus-25	source	3032333912	15634872071	487	0

Retrieve complete call detail records

NexSRS CDR Fields

The following table describes all the fields in a NexSRS call detail record. NexSRS CDRs are written to a tab delimited text file. Each CDR is separated by a line space.

CDR Field	Description
Date/Time	• For internal CDR: date/time when call authorization request is received by the NexSRS server. • For source CDR: date/time when source usage indication message is received by the NexSRS server. • For destination CDR: date/time when destination usage indication message is received by the NexSRS server.
Reporting Device Name	The device that sends (reports) the call authorization request or the usage indication message to the NexSRS server.
Product Name	The Product used by the NexSRS Server to determine the call destinations. This field is only populated in internal CDRs.
Source Device Name	The device that originates the call.
Source Customer	The Customer associated with the source device.
Destination Device Name	The device that terminates the call.
Destination Customer	The Customer associated with the destination device.
Calling Number Authorized	In the source CDR, this value is the original Calling Number reported in the OSP AuthorizationRequest (before any number translations). In the internal and destination CDRs, this value is the translated Calling Number returned to the source device in the OSP AuthorizationResponse (after all number translations).
Called Number Authorized	In the source CDR, this value is the original called number reported in the OSP AuthorizationRequest (before any number translations). In the internal and destination CDRs, this is the translated Called Number returned to the source device in the OSP AuthorizationResponse (after all number translations).
CDR Type	• internal (CDRs generated by the NexSRS server at the time the OSP authorization request is received) • source (originating) – reported by the source device. • destination (terminating) – reported by the destination device.
Callid	A base64-encoded value created/passed by the source device; its use varies depending on the call signaling protocol in use (e.g., H.323, SIP, etc.), but in general, the source device may use this value to distinguish individual calls. If the source device sends one Callid in the authorization request, this Callid should appear in the internal CDR generated by the NexSRS server for each destination returned to the source device. If the source device sends multiple Callid's, each internal CDR will include a different Callid for each destination returned by the

CDR Field	Description
	NexSRS server to the source device. The Callid reported in the source and destination CDRs will be the values reported by the Source and Destination devices.
Transaction ID	A unique transaction ID generated by the NexSRS Server for each call authorization request; this value can be used to match all CDRs generated for a single call.
Duration	Call duration in seconds as reported by Source and Destination devices; this value is zero in the internal CDR.
Count	An OSP UsageIndication message can contain multiple CDRs. The count value for a CDR is its numbered order in a UsageIndication message. For example, if a source UsageIndication message contains three CDRs, count = 1 for the first CDR, count = 2 for the second CDR and count = 3 for the third CDR.
Total	An OSP UsageIndication message can contain multiple CDRs. The total value is the total number of CDRs in a UsageIndication message.
TC Code	OSP Termination Cause Code indicating the reason why the call was terminated. Please see Appendix 3 for a complete list of OSP Termination Cause Codes.
Call Start Time	The time when the call started as reported by the source and destination gateways.
Called Number (Cld Num) Requested	The called number in the authorization request received from the source device. (This value is only valid in the internal CDR.)
Cld Num after In-Prefix	The called number after the Inbound prefix is removed. (This value is only valid in the internal CDR.)
Cld Num after Pre-routing Number Trans.	The called number after pre-routing number translation. (This value is only valid in the internal CDR.)
Cld Num after Post-routing Number Trans.	The called number after post-routing number translation. (This value is only valid in the internal CDR.)
Calling Number (Clg Num) Requested	The calling number in the authorization request received from the source device. (This value is only valid in the internal CDR.)
Clg Num after pre-routing Number Trans.	The calling number after pre-routing number translation. (This value is only valid in the internal CDR.)
Call End Time	The time when the call ended as reported by the source and destination gateways.
Call Alert Time	The time when the call alerting or ringing began as reported by the source and destination gateways.
Post Dial Delay	The post dial delay as reported by the source device.
Source Network ID	Trunk group for the source device
Destination Network ID	Trunk group for the destination device
Proxy IP Address	When a signaling proxy performs LookAhead routing, this field

CDR Field	Description
	holds the IP address of the proxy.
Call Connect Time	Time stamp of when the call connected (Q931 connect for H.323 or 200 OK for SIP).

Summary of NexSRS Called Number Values

The NexSRS server records five different values for the called number in the internal CDR to provide a complete audit trail of how the called number is manipulated by the NexSRS number translation features. The table below summarizes the sequential changes to the called number by functions in the NexSRS server.

Input ➔	NexSRS Function ➔	Output
Called Number (Cld Num) Requested	If an In-Prefix is defined for the source device and present in the called number, remove the In-Prefix from the called number. Otherwise, block the call.	Cld Num after In-Prefix
Cld Num after In-Prefix	Perform pre-routing number translation	Cld Num after Pre-routing Number Translation
Cld Num after Pre-routing Number Translation	Routing: Determine IP addresses of destination devices that can complete a call to the called number.	Cld Num after Pre-routing Number Translation
Cld Num after Pre-routing Number Translation	Perform post-routing number translation	Cld Num after Post-routing Number Translation
Cld Num after Post-routing Number Translation	If an Outbound prefix is defined for the destination device, add the Out-Prefix to the called number	Called Number Authorized

Summary of NexSRS Calling Number Values

The NexSRS server records three different values for the calling number in the internal CDR to provide a complete audit trail of how the calling number is manipulated by the NexSRS number translation features. The table below summarizes the sequential changes to the calling number by functions in the NexSRS server.

Input ➔	NexSRS Function ➔	Output
Calling Number (Clg Num) Requested	Perform pre-routing number translation	Clg Num after pre-routing Number Trans.
Clg Num after pre-routing Number Trans.	Routing: Determine IP addresses of destination devices that can complete a call to the called number.	Clg Num after pre-routing Number Trans.
Clg Num after pre-routing Number Trans.	Perform post-routing number translation	Calling Number Authorized

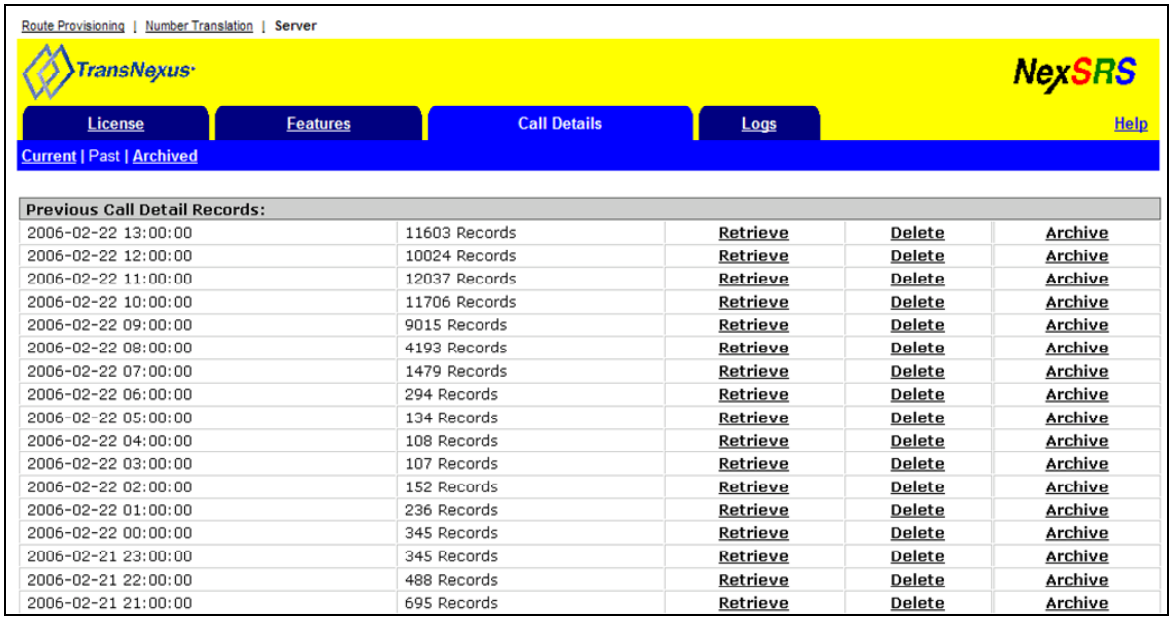
CDRs from a Stateless Device

Call Detail Records from devices which do not maintain call state, such as a SIP proxy, will report at least two source and destination CDRs for each call. A stateless SIP proxy will report a start CDR when the call connects and a stop CDR when the call ends. If a SIP proxy has to retry multiple destinations to complete the call, the SIP proxy will report multiple start CDRs – one for each call attempt. If the call fails, the TC code in the source start CDR will be the SIP error code returned. A start CDR will report CallStartTime, CallAlertTime and CallConnectTime.

A stop CDR, either source or destination, will always indicate a successful call. A stop CDR will report CallEndTime and the call Duration (CallEndTime less CallStartTime). Stop CDRs have a TC code of 10016 so they can be distinguished from CDRs from stateful devices which report a TC code of 1016 or 16 for a successful call.

Past CDRs

CDR files are created periodically by the NexSRS server. The frequency of CDR file creation is dertermined by the `CDR_FILE_TIME_SPAN` parameter defined in the start_osp_server.sh script (see Chapter 2). This parameter is defined in minutes and may be set to 1, 2, 3, 4, 5, 6, 10, 12, 15, 20, 30 or 60 minutes. For example, if `CDR_FILE_TIME_SPAN=5`, then a CDR file will be cut every five minutes. Past CDR files are stored in the `$NexSRS_HOME/xitami/osp_ui/cdrs` directory. The naming convention for NexSRS server CDR files is `cdr_DATE__TIME.cdr` where DATE and TIME are the start time of the cdr file. A list of these past CDR files, and the number of records they contain, may be viewed from the Past page under the **Call Details** tab.


Route Provisioning | Number Translation | Server
TransNexus
NexSRS
License
Features
Call Details
Logs
Help
Current | Past | Archived
Previous Call Detail Records:
2006-02-22 13:00:00 11603 Records Retrieve Delete Archive
2006-02-22 12:00:00 10024 Records Retrieve Delete Archive
2006-02-22 11:00:00 12037 Records Retrieve Delete Archive
2006-02-22 10:00:00 11706 Records Retrieve Delete Archive
2006-02-22 09:00:00 9015 Records Retrieve Delete Archive
2006-02-22 08:00:00 4193 Records Retrieve Delete Archive
2006-02-22 07:00:00 1479 Records Retrieve Delete Archive
2006-02-22 06:00:00 294 Records Retrieve Delete Archive
2006-02-22 05:00:00 134 Records Retrieve Delete Archive
2006-02-22 04:00:00 108 Records Retrieve Delete Archive
2006-02-22 03:00:00 107 Records Retrieve Delete Archive
2006-02-22 02:00:00 152 Records Retrieve Delete Archive
2006-02-22 01:00:00 236 Records Retrieve Delete Archive
2006-02-22 00:00:00 345 Records Retrieve Delete Archive
2006-02-21 23:00:00 345 Records Retrieve Delete Archive
2006-02-21 22:00:00 488 Records Retrieve Delete Archive
2006-02-21 21:00:00 695 Records Retrieve Delete Archive

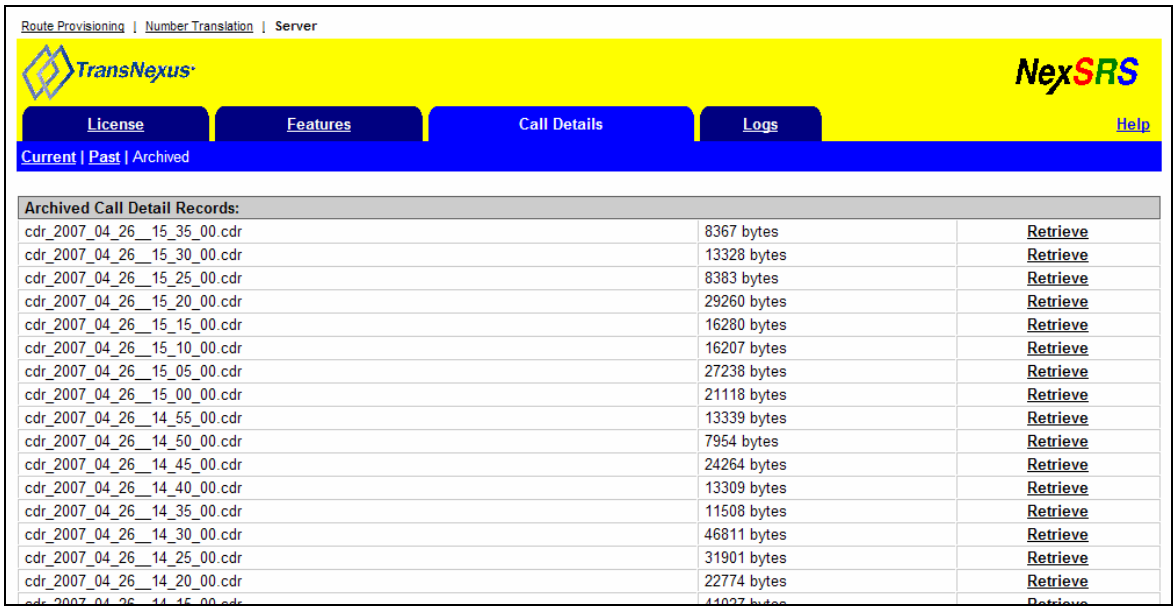
Past CDR files may be downloaded to a local computer by clicking on the **Retrieve** link which will prompt the user to open the file or save it to the local computer. Past CDR files may be archived locally on the NexSRS server by clicking on the **Archive** link which moves the cdr file

to the `$NexSRS_HOME/xitami/osp_ui/cdrs/ARCHIVE` directory. Past CDR files may also be deleted from the NexSRS server by clicking the **Delete** link.

TransNexus recommends that NexSRS server CDR files not be manually deleted. A much more reliable operating procedure is to use NexOSS to automatically copy NexSRS server cdr files to a central mediation directory and then archive NexSRS server cdr files locally on the NexSRS server. This process ensures there are at least two copies of each cdr file through the billing cycle. Please see the NexOSS User Manual for more information on the `cdr_pull` and `cdr_file_audit` functions.

Managing Archived CDR Files

When a CDR file is retrieved by the NexOSS cdr_pull application, the CDR file is removed from the Past user interface page and a copy of the CDR file is archived in the `$NexSRS_HOME/xitami/ui/cdrs/ARCHIVE` directory of the NexSRS server. Archived CDR files may be viewed and retrieved from the NexSRS server from the Archived user interface page as shown below.



Archived Call Detail Records:		
cdr_2007_04_26__15_35_00.cdr	8367 bytes	Retrieve
cdr_2007_04_26__15_30_00.cdr	13328 bytes	Retrieve
cdr_2007_04_26__15_25_00.cdr	8383 bytes	Retrieve
cdr_2007_04_26__15_20_00.cdr	29260 bytes	Retrieve
cdr_2007_04_26__15_15_00.cdr	16280 bytes	Retrieve
cdr_2007_04_26__15_10_00.cdr	16207 bytes	Retrieve
cdr_2007_04_26__15_05_00.cdr	27238 bytes	Retrieve
cdr_2007_04_26__15_00_00.cdr	21118 bytes	Retrieve
cdr_2007_04_26__14_55_00.cdr	13339 bytes	Retrieve
cdr_2007_04_26__14_50_00.cdr	7954 bytes	Retrieve
cdr_2007_04_26__14_45_00.cdr	24264 bytes	Retrieve
cdr_2007_04_26__14_40_00.cdr	13309 bytes	Retrieve
cdr_2007_04_26__14_35_00.cdr	11508 bytes	Retrieve
cdr_2007_04_26__14_30_00.cdr	46811 bytes	Retrieve
cdr_2007_04_26__14_25_00.cdr	31901 bytes	Retrieve
cdr_2007_04_26__14_20_00.cdr	22774 bytes	Retrieve

Archived CDR files will accumulate in the `ARCHIVE` directory of the NexSRS server until they are deleted Please see the Utilities section of Chapter 7 for information on scripts which automate the process of compressing archived CDR files and deleting old files in the `/ARCHIVE` directory. It is critical for archived CDR files to be deleted periodically to free up hard disk space on the NexSRS server. The NexSRS server will shutdown if free disk space falls below the `AVAILABLE_DISK_SPACE_REQUIRED_THRESHOLD` limit (see Chapter 2 on NexSRS server Configuration). For planning purposes, NexSRS server operators should allocate 2,000 bytes of CDR storage for each call. For example, if a NexSRS server handles 100,000 calls per day, then sufficient hard disk space must be maintained on the NexSRS server to handle 200 MB of new CDRs each day.

Chapter 9

User Administration

The NexSRS web server may be configured to limit user access to the NexSRS user interface based on User Name, Password and the IP address of the machine accessing the user interface. The configuration file for administering web user access to the NexSRS server is the xitami.aut file is located in the $NexSRS_HOME/xitami directory. The Xitami.aut file can be modified using a text editor. Once the xitami.aut file has been modified and saved to the $NexSRS_HOME/xitami directory, the configuration changes will take effect when the Xitami web server is restarted.

Below is a list of web directories which may be password protected to limit user access to the NexSRS server web user interface.

1. /ui – contains HTML and script files.
2. /log – contains NexSRS log files.
3. /CDRs – contain Call Detail Record files
4. /cgi-bin – a single path to protect all view and update resources (excluding log and CDR files)
5. /cgi-bin/data – a single path to protect all view requests (excluding log and CDR files)
6. /cgi-bin/action – a single path to protect all update requests (excluding log and CDR files)

The Xitami.aut file includes a default configuration for View-Only and Administrative users. For customizing user access policies, see the detailed NexSRS unter interface site map at the end of this chapter for information on password proecting specific pages. Instructions for modifying the xitami.aut file to manage user access are provided within the xitami.aut file. The full contents of the xitami.aut file, including instructions, are shown below.

```
########################################################################
##                                                                    #
##                NexSRS Web Server User Administration               #
##                             XITAMI.AUT                             #
##                                                                    #
########################################################################
```

```
##
## This is the configuration file for the web server packaged with the
## NexSRS server.  By editing this file, the NexSRS Administrator can
## limit and control user access to the NexSRS web user interface.
##
## Rows that begin with ## are comment rows.  Comment rows and blank
## rows have no impact the web server configuration or user access.
##
## This configuration file is divided into four main sections:
##
## 1 DEFINITIONS       Definition of terms
##
## 2 INSTRUCTIONS      How to configure user access
##
## 3 USER ADMINISTRATION ACCESS CONTROL
##    3.1 VIEW ONLY    Users may view all data and download log file.
##
##    3.2 ADMIN        Full access to change data - restore routing and
##                     number translation configuration files .
##                     retrieve, archive and delete CDR files.
##                     NexOSS must have Admin access.
##
## 4 FIXED PARAMETERS  Required configuration parameters that should
##                     not be changed.
##
#########################################################################
#########################################################################
## DEFINITIONS
## ===========
##
## Realm
## =====
## A realm defines a web page, a directory of web pages, a link or
## an action that a user may request from the web server.
##
## User Name
## =========
## User names are a single string of characters or numbers; they
## ARE NOT case sensitive.  That is to say, .john. is the same as
## .JoHn..
##
## User Password
## =============
## User passwords are a single string of characters enclosed in
## double quotes, for example, "password".  Passwords may contain
## any printable character except single quotes (') and double
## quotes (").  The password with nothing between the double quotes
## "" means a password is not required for that user.  The password
## "-" means no access.  Passwords ARE case sensitive.  To enable
## global access use all=* to allow access to all users.  User name
## and password examples:
##
## User1="we1c0me"   User name and password with alpha-numeric
##                   characters.
```

```
## User2="WE1C0ME"   Passwords for User1 and User 2 are different
##                   since the password value is case sensitive.
## User3=""          No password is required for User3.
## User4="-"         No access
## all=*             Allow access to all users.
##
## Webmask
## =======
## The Webmask parameter defines, by IP address, what machines may
## access the NexSRS server web user interface.  Webmask= is followed
## by a pattern (enclosed in double quotes " ") which specifies valid
## addresses.  A pattern may be a complete IP address or a partial IP
## address with '*' replacing part of the end of the address.  For
## example, the pattern "250.12.13.*" restricts access only to clients
## with IP addresses starting with .250.12.13'.  A pattern "*" matches
## all addresses.  One Webmask parameter may configured for each realm
## or privilege.  The Webmask may contain a list of IP addresses, each
## separated by a comma.  If no patterns are entered after Webmask=,
## the corresponding realm or privilege is not accessible to any users.
##
## Examples:
## Webmask="172.23.6.1"  A user on a machine with IP address 172.23.6.1
##                       may be granted access.
## Webmask="172.*"       A user on a machine with IP address beginning
##                       with 172 may be granted access.
## Webmask="172.*,159.*" This example is a comma separated list of IP
##                       addresses.  A user on a machine with IP
##                       address
##                       beginning with 172 or 159 may be granted access.
## Webmask="local"       Only a user on the local system may be granted
##                       access.
## Webmask="*"           A user from any IP address may be granted
##                       access.
## Webmask=              No user, from any IP address, may be granted
##                       access.
##
## If the NexSRS administrator wants to use the same Webmask IP
## addresses repetitively for different realms, then the webmask
## parameter may be pointed to a file which holds the IP addresses.
## For example: Webmask=@iplist.txt where iplist.txt file includes a
## list of IP addresses separated by spaces, commas or blank lines.
## The iplist.txt file must be stored in the $NexSRS_HOME/xitami directory.
##
#########################################################################
#########################################################################
## INSTRUCTIONS
## ============
##
## To change user access privileges, this file must be edited and
## saved in the $NexSRS_HOME/xitami directory.  The xitami web server
## must be restarted for the configuration changes to take affect.
##
## Managing user access requires configuration of four parameters.
## 1. The actual realm to be protected.
```

```
## 2. The name assigned to the realm.
## 3. User names and passwords which may access the realm.
## 4. Webmask of IP addresses that may access the realm.
##
## Below is an example configuration granting User1 access to
## download the NexSRS log file.
##
## /log/nexus.log
## Realm="NexSRS"
## Webmask="196.*"
## User1="Password of User1"
##
## This configuration enables User1 to download the NexSRS log file
## from any machine with IP address beginning with 196.  This example,
## presents one Webmask and one User Name="Password".  However, there
## may be one or many User Name="Password" values and none or many
## Webmask values.  These lines must be uncommented (remove ## at the
## beginning of each line) in order for the configuration to take
## affect.
##
## The next section provides a configuration framework for managing
## users with different levels of access.  The NexSRS realms are
## predefined and need only be uncommented.  In addition, user names
## with passwords and webmasks must be added.  For a complete listing
## of all NexSRS realms, please see the NexSRS User Manual.  The NexSRS
## Administrator is not restricted to using the suggested predefined
## configurations.  This file may be re-configured to meet any
## customized user access policy.
##
## Xitami.aut may be customized to implement any User Adminstration
## security policy.  For expert help creating a customized Xitami.aut
## configuration contact support@TransNexus.com.
##
#########################################################################
#########################################################################
##
## USER ADMINISTRATION
## ===================
##
## VIEW-ONLY USERS
## To enforce uniform view-only privileges, the same set of users
## and webmasks must be configured for the web directories below.
## The users configured in the realms below may:
##   - View all data on all pages of the NexSRS server user interface
##   - Download the Number Translation configuration file.
##   - Download the Log file.
##   - Retrieve CDR files.
##
## Access to view all data.
##[/cgi-bin/data]
##Realm="NexSRS"
##Webmask="comma separated IP addresses or @iplist.txt"
##View_Only="ViewOnly_Password"
##
```

```
## NOTE! Admin users must also be configured for access to the
## directories below.
##
## Access to view HTML banner for each user interface web page.
##[/ui]
##Realm="NexSRS"
##Webmask="comma separated IP addresses or @iplist.txt"
##View_Only="ViewOnly_Password"
##Admin="Admin_Password"
##
## Access to download the NexSRS log files.
##[/log]
##Realm="NexSRS"
##Webmask="comma separated IP addresses or @iplist.txt"
##View_Only="ViewOnly_Password"
##Admin="Admin_Password"
##
## Note! Admin and NexOSS users must also be configured for access
## to the directories below.
##
## Access download CDR files.
##[/CDRs]
##Realm="NexSRS"
##Webmask="comma separated IP addresses or @iplist.txt"
##View_Only="ViewOnly_Password"
##Admin="Admin_Password"
##NexOSS="NexOSS_Password"
##
##
## ADMIN USERS
## Admin users and NexOSS have complete access to all NexSRS functions.
##    - Clear the NexSRS Log file
##    - Enable/disable NexSRS features
##    - Change License
##    - Backup and restore routing configuration file
##    - Backup and Restore number translation configuration files
##    - Backup and Route Plan Assignments
##    - Download, delete or archive CDR files
##
## NOTE: Admin users must also be granted access to the /ui, /CDRs
## and /log paths configured in the VIEW-ONLY section above.
##
## Access to view and change all data.
##[/cgi-bin]
##Realm="NexSRS"
##Webmask="comma separated IP addresses or @iplist.txt"
##Admin="Admin_Password"
##NexOSS="NexOSS_Password"
##
#############################################################################

#############################################################################
##
## Do not password protect the applet library.
```

```
##
[/ui/lib]
all=*
##
## Also, the following paths should not be password protected.
## /osp – OSP messages
## /tep and /scep – enrollment messages
##
#########################################################################

#########################################################################
##
## 4 FIXED PARAMETERS
## ==================
##
## The parameters in this section should not be changed.
##
## Xitami Administration Privilege
## ==============================
## Do not list any users or Webmask patterns.
##
[/admin]
Realm="Admin"
Admin="-"
Webmask=
#
######################## End of Xitami.aut ##############################
```

Once the Xitami.aut file has been edited and saved in the $NexSRS_HOME/xitami directory. The NexSRS web server must be restarted for the new configuration to take affect. The NexSRS Administrator does not need to stop and restart the NexSRS server to restart Xitami. Instead the NexSRS Administrator may stop the xitami process (xitami is the NexSRS web server) and allow Nexus Watch Dog to restart Xitami.

Chapter 10

Troubleshooting

Log messages on the NexSRS server are the best resource for trouble shooting any problems which may occur from the configuration or operation of the NexSRS server. This section provides a listing of common log messages, an explanation and a remedy for error conditions. The log messages are grouped into the following categories.

- Normal Operation
- Routing Configuration
- Features Configuration
- Device Enrollment
- CDR Management
- License
- 400 Error Codes – Bad Request
- 401 Error Codes – Not Authorized
- 404 Error Codes – Route Not Found
- 503 Error Codes – Service Not Available
- OSP Client Errors
- ExtraNet

If problems with the NexSRS server occur, first refer to the log messages. If the problem cannot be resolved run the script `collect_debug_info.sh` script and e-mail the output from this script to support@transnexus.com with a description of the problem.

Normal Operation

Log Message (heartbeat)

```
Call Attempts: <171> for past minute, <8366> for past hour
```

Explanation

This is a "heartbeat" log message which indicates every minute that the NexSRS server is operating. Every minute the NexSRS Server logs a message of the number of call attempts during the previous minute and previous hour.

Normal Operation

Log Message (heartbeat)

```
Call Attempts: <84> for past minute, <2288> for past hour, <3520> for
past hour extrapolated.
```

Explanation

This is a "heartbeat" message from the first one hour after starting the NexSRS server. The server is licensed by number of call attempts allowed per hour (Busy Hour Call Attempts, BHCA). During the first hour of operation, the total number of calls per hour is extrapolated.

Log Message (disk space warning)

```
WARNING: Available disk space is 118 MB.
WARNING: NexSRS server will shutdown when available disk space falls
below 15 MB.
Increase available disk space in the partition where the NexSRS server
is installed.
```

Reason for log message

The NexSRS server issues a warning when the available disk space falls below the `AVAILABLE_DISK_SPACE_WARNING_THRESHOLD`. The default value for this variable is 1000 MB. The NexSRS server will shutdown when the available disk space falls below the `AVAILABLE_DISK_SPACE_REQUIRED_THRESHOLD`. The default value is 15 MB. Both of these variables are configured in the `start_osp_server.sh` script.

Remedy

1. Increase available disk space on the NexSRS server by either removing or compressing CDR files, or
2. Decrease the `AVAILABLE_DISK_SPACE_WARNING_THRESHOLD` value in the `start_osp_server.sh` file. The NexSRS server must be restarted for changes to the `start_osp_server.sh` to take affect.

Log Message (alphabet characters ignored)

```
Translated DestinationInfo from <911884267106A> to <911884267106>
```

Explanation

The NexSRS server will route calls based on any combination of the following thirteen characters: integers 0-9, * (asterisk), # (hash) and , (comma). The NexSRS server will also accept destination numbers with alphabet characters, but the alphabet characters will be ignored. The log message above is an example of a destination number that includes A (an alphabet character), which is removed before the NexSRS server routes the call.

Normal Operation
Log Message (clear Log) `Event log cleared by user “UserName”` **Explanation** `“UserName”` has cleared all records in the nexus.log file using the Clear Log link on the Logs page, under the **Server** tab, of the NexSRS server user interface.

Routing Configuration

Routing Configuration

Log Messages (add Customer)
`Customer 'Test Group' added by user “UserName”`
`Starting to copy configuration.`
`Starting to delete configuration.`
`Finished deleting configuration.`
`Finished copying configuration.`
`Successfully pushed configuration changes to the server`
 	
Explanation
These log messages confirm that Customer `‘Test Group’` was successfully added to the NexSRS server configuration.

Log Messages (delete Customer)
`Customer 'Test Group' and all of its devices and routes deleted by user “UserName”`
`Starting to copy configuration.`
`Starting to delete configuration.`
`Finished deleting configuration.`
`Finished copying configuration.`
`Successfully pushed configuration changes to the server`
 	
Explanation
These log messages confirm that Customer `‘Test Group’` was successfully deleted from the NexSRS server configuration.

Log Messages (enable Customer)
`Customer 'Test Group' enabled by user “UserName”`
`Starting to copy configuration.`
`Starting to delete configuration.`
`Finished deleting configuration.`
`Finished copying configuration.`
`Successfully pushed configuration changes to the server`
 	
Explanation
These log messages confirm that Customer `'Test Group'` was enabled by user `“UserName”` checking the May originate box for on the **Customers** page of the NexSRS server user interface.

Routing Configuration

Log Messages (disable Customer)

```
Customer 'Test Group' disabled by user "UserName"
Starting to copy configuration.
Starting to delete configuration.
Finished deleting configuration.
Finished copying configuration.
Successfully pushed configuration changes to the server
```

Explanation

These log messages confirm that Customer `'Test Group'` was disabled by user `"UserName"` un-checking the May originate box for on the **Customers** page of the NexSRS server user interface.

Log Messages (add Device failure)

```
Failed to resolve IP address for <TestDevice>
Unable to add device
```

Reason for Log Message

These log messages indicate that device `<TestDevice>` could not be added to the NexSRS server configuration. The reason the device could not be added was because the device name could no be resolved by a DNS query to determine the IP address of the device.

Remedy

1. Enter the IP address of the device to be added, not the device name, or
2. Make certain that the device name can resolved to an IP address by a DNS query.

Log Messages (add Device)

```
Device '12.34.56.78' added to Customer 'Destinations' by user
"Username"
Starting to copy configuration.
Starting to delete configuration.
Finished deleting configuration.
Finished copying configuration.
Successfully pushed configuration changes to the server
```

Explanation

Device `'12.34.56.78'` has been successfully added to the `'Destinations'` Customer in the NexSRS server configuration by `"UserName"`.

Log Messages (enable Device)

```
Device '12.34.56.78' enabled by user "UserName"
Starting to copy configuration.
Starting to delete configuration.
Finished deleting configuration.
Finished copying configuration.
Successfully pushed configuration changes to the server
```

Routing Configuration
Explanation "UserName" has enabled device '12.34.56.78' to terminate calls by checking the May terminate status box on the **Devices** page of the NexSRS server user interface.
Log Messages (disable Device) Device '12.34.56.78' disabled by user "UserName" Starting to copy configuration. Starting to delete configuration. Finished deleting configuration. Finished copying configuration. Successfully pushed configuration changes to the server **Explanation** "UserName" has disabled device '12.34.56.78' from terminating calls by un-checking the May terminate status box on the **Devices** page of the NexSRS server user interface.
Log Messages (delete Device) Device '12.34.56.78' deleted by user "UserName" Starting to copy configuration. Starting to delete configuration. Finished deleting configuration. Finished copying configuration. Successfully pushed configuration changes to the server **Explanation** Device '12.34.56.78' has been deleted from the NexSRS server configuration by "UserName".
Log Message (add route) Route to '1404' for Customer 'Test Group' added by user "UserName" Starting to copy configuration. Starting to delete configuration. Finished deleting configuration. Finished copying configuration. Successfully pushed configuration changes to the server **Explanation** The route from any 'Test Group' device to '1404' has been added to the NexSRS server configuration by "UserName".
Log Message (edit route) Route to '1404' for Customer 'Test Group' deleted by user Route to '1404526' for Customer 'Test Group' added by user "UserName" Starting to copy configuration. Starting to delete configuration. Finished deleting configuration. Finished copying configuration. Successfully pushed configuration changes to the server

Routing Configuration

Explanation
The route from any `'Test Group'` device to `'1404'` has been changed to `'1404526'` in the NexSRS server configuration by `"UserName"`.

Log Message (delete route)
```
Route to '1404526' for Customer 'Test Group' deleted by user
Starting to copy configuration.
Starting to delete configuration.
Finished deleting configuration.
Finished copying configuration.
Successfully pushed configuration changes to the server
```

Explanation
The route from any `'Test Group'` device to `'1404526'` has been deleted from the NexSRS server configuration by `"UserName"`.

Log Messages (Restore configuration file)
```
Created Configuration Back Up file <nexus_cfg_2002-12-27-13-47-08.bu>
Successfully started TEP and CEP processes
Turned Authorization request on
Turned Usage Indication Collection on
Starting to copy configuration.
Starting to delete configuration.
Finished deleting configuration.
Finished copying configuration.
Successfully pushed configuration to the Server
New configuration is in effect
Destructing old GroupManager
Starting to delete configuration.
Finished deleting configuration.
```

Explanation
These log messages indicate that an NexSRS server configuration file has been successfully restored using the Restore link on the Backup page, under the **Server** tab, of the NexSRS Server user interface.

Log Messages (Invalid configuration item)
```
Created Configuration Back Up file <nexus_cfg_2002-12-31-18-51-55.bu>
Line number: 1 contains invalid configuration item: Created
Failed to process the configuration file
This configuration will not be pushed to the server
Uploaded configuration file has errors, rolling back to the original
configuration.
```

Reason for log message
Line number 1 of the configuration file contains an item that is not part of the normal configuration file syntax. In this example, the line `"#Created by NexSRS Server on Tue Dec 31 03:00:07 2002`", which is a standard comment line at the top of the configuration file, was uncommented before being restored to the NexSRS server.

Routing Configuration

Remedy
Using a text editor remove, or comment out, any extra text in the configuration that is unrelated to the NexSRS server configuration. Or restore a different configuration file that is a valid NexSRS server configuration file.

Log Messages (Restore Configuration file failed – invalid license)

```
Created Configuration Back Up file <nexus_cfg_2002-12-31-18-18-33.bu>
Failed to extract Signature element
Rolling back to the default values
Failed to set the license, default values will be used
Successfully started TEP and CEP processes
Turned Authorization request on
Turned Usage Indication Collection on
Starting to copy configuration.
Starting to delete configuration.
Finished deleting configuration.
Finished copying configuration.
Successfully pushed configuration to the Server
New configuration is in effect
Destructing old GroupManager
Starting to delete configuration.
Finished deleting configuration.
```

Reason for log messages
The configuration file restored to the NexSRS server, using the Restore configuration from: link on the Backup page under the **Server** tab, has an invalid license. The licensed parameters of the NexSRS server have been reset using default license parameters.

Remedy
Install a new NexSRS server license. Click on the Upgrade with new License Key link on the License page under the **Server** tab of the NexSRS server user interface. The **Enter New License Key:** window will appear. Cut and paste new license into the **Enter New License Key:** window. Or, use the Restore configuration from: link on the Backup page, under the **Server** tab of the NexSRS server to restore a configuration file that contains a valid license.

Log Messages (config_push)

```
Created Configuration Back Up file <nexus_cfg_2002-12-25-23-00-01.bu>
Successfully started TEP and CEP processes
Turned Authorization request on
Turned Usage Indication Collection on
Starting to copy configuration.
Starting to delete configuration.
Finished deleting configuration.
Finished copying configuration.
Successfully pushed configuration to the Server
New configuration is in effect
Destructing old GroupManager
Starting to delete configuration.
```

Routing Configuration

`Finished deleting configuration.`
`Status: 200`
 	
Explanation
These log messages indicate a successful config_push operation from the NexOSS Manager. An updated nexus.cfg configuration file has been successfully pushed to the NexSRS server from the NexOSS Manager.

Features Configuration

Features Configuration
Log Message (enable enrollment) `Device enrollment enabled by user "Username"` **Explanation** `"UserName"` has enabled the enrollment feature of the NexSRS server on the Features page, under the **Server** tab, of the user interface.
Log Message (disable enrollment) `Device enrollment disabled by user "UserName"` **Explanation** `"UserName"` has disabled the enrollment feature of the NexSRS server on the Features page, under the **Server** tab, of the user interface.
Log Message (enable routing) `Routing enabled by user "UserName"` **Explanation** `"UserName"` has enabled the routing feature of the NexSRS server on the Features page, under the **Server** tab, of the user interface.

Log Message (disable routing)
`Turning Routing Off, there are <38> transactions in progress for <osp>`
`Turned Routing Off for <osp>`
`Routing disabled by user "UserName"`
 	
Explanation
`"UserName"` has disabled the routing feature of the NexSRS server on the Features page, under the **Server** tab, of the user interface. The NexSRS server will not respond to OSP *AuthorizationRequests* and will log the following message.
 	
`(Error 503 Service Not Available) AuthRequest denied, feature is disabled, from Device <DeviceName>`

Features Configuration

Log Message (disable CDR collection)
`Turning CDRCollection Off, there are <102> transactions in progress for <osp>`
`Turned CDRCollection Off for <osp>`
`CDR collection disabled by user "UserName"`
 	
Explanation
`"UserName"` has disabled the CDR collection feature of the NexSRS server on the Features page, under the **Server** tab, of the user interface. The NexSRS server will reject any OSP UsageIndication messages it receives and will log the following message.
 	
`(Error 503 Service Not Available) UsageInd message rejected; Collect CDRs feature is disabled, from Device <DeviceName>`

Log Message (enable SSL)
`Secure Sockets Layer enabled by user "UserName"`
 	
Explanation
`"UserName"` has enabled the Secure Sockets Layer feature of the NexSRS server on the Features page, under the **Server** tab, of the user interface. If any OSP client device sends an OSP *AuthorizationRequest* or *UsageIndication* message to the NexSRS server using http (non-SSL, port 1080) the NexSRS server the message will rejected and the NexSRS server will log the following message.
 	
`(Error 503 Service Not Available) SSL is enabled, OSP requests are forbidden on the non SSL port at this time.`

Log Message (disable SSL)
`Secure Sockets Layer disabled by user "UserName"`
 	
Explanation
`"UserName"` has disabled the Secure Sockets Layer feature of the NexSRS server on the Features page, under the **Server** tab, of the user interface. If any OSP client device sends an OSP *AuthorizationRequest* or *UsageIndication* message to the NexSRS server using SSL (port 1443) the NexSRS server the message will rejected and the NexSRS server will log the following message.
 	
`(Error 503 Service Not Available) SSL is disabled, OSP requests are forbidden on the SSL port at this time.`

Device Enrollment

Device Enrollment

Device Enrollment
Log Message (enrollment successful) `Device <DeviceName> has enrolled` **Explanation** The Device `<DeviceName>` has successfully enrolled itself with the NexSRS server.
Log Message (enrollment failed) `Device <DeviceName> is not defined, TEP GetClientCert request is denied!` **Reason for log message** Device `<DeviceName>` is not defined in the NexSRS server. The NexSRS server will not enroll an unknown device. **Remedy** Add the Device `<DeviceName>` to the NexSRS server configuration.
Log Message (SCEP enrollment failed) `(Error 400 Bad Request) Not supported message from Device <DeviceName>` **Reason for log message** An enrollment request from Device `<DeviceName>` using SCEP enrollment protocol was incorrectly addressed to the NexSRS server enrollment url followed by /OSP. **Remedy** In the OSP client (Device `<DeviceName>`), change the address of the NexSRS server enrollment url to end with /SCEP instead of /OSP.
Log Message (TEP enrollment failed) `(Error 400 Bad Request) Not supported message from Device <DeviceName>` **Reason for log message** An enrollment request from Device `<DeviceName>` using TEP enrollment protocol was incorrectly addressed to the NexSRS server enrollment url followed by /OSP. **Remedy** In the OSP client (Device <DeviceName>), change the address of the NexSRS server enrollment url to end with /TEP instead of /OSP.
Log Message (TEP enrollment failed) `Failed to parse an empty query string` **Reason for Log Message** An enrollment request from a device using TEP enrollment protocol was addressed to the enrollment url of the NexSRS server followed by /SCEP. **Remedy** In the OSP client, change the address of the NexSRS server enrollment url to end with /TEP instead of /SCEP.

Device Enrollment
Log Message (TEP enrollment failed) `Unrecognized script: /cgi-bin` **Reason for Log Message** Enrollment Request from a device using TEP enrollment protocol was addressed to the enrollment url of the NexSRS server followed by /cgi-bin. **Remedy** In the OSP client, change the address of the NexSRS server enrollment url to end with /TEP instead of /cgi-bin.
Log Message (enable enrollment) `Device enrollment enabled by user "UserName"` **Explanation** `"UserName"` has enabled the enrollment feature of the NexSRS server on the Features page, under the **Server** tab, of the user interface.
Log Message (disable enrollment) `Device enrollment disabled by user "Username"` **Explanation** `"UserName"` has disabled the enrollment feature of the NexSRS server on the Features page under the **Server** tab, of the user interface.

CDR Management

CDR Management
Log Message (delete CDR file) `CDR file './xitami/osp_ui/cdrs/cdr_2002_11_18__15_00_00.cdr' deleted by user "UserName"` **Explanation** `"UserName"` has deleted the cdr file `cdr_2002_11_18__15_00_00.cdr` from the NexSRS server using the Delete link on the Past CDRs page, under the **Call Details** tab, of the NexSRS server user interface.
Log Message (archive CDR file) `CDR file './xitami/osp_ui/cdrs/cdr_2002_11_18__20_00_00.cdr' archived by user "UserName".` **Explanation** `"UserName"` has archived the cdr file `cdr_2002_11_18__20_00_00.cdr` from the NexSRS server using the Archive link on the Past page, under the **Call Details** tab, of the NexSRS server user interface.

CDR Management

Log Messages (cdr_pull)

```
Found a past CDR File </cdrs/cdr_2002_12_25__21_00_00.cdr>
CDR file './xitami/osp_ui/cdrs/cdr_2002_12_25__21_00_00.cdr' archived
by user "UserName"
No past CDR file name was found, try again later
```

Explanation

These log messages indicate a successful cdr_pull operation from the NexOSS Manager. The NexOSS Manager identified the `cdr_2002_12_25__21_00_00.cdr` CDR file on the NexSRS server, retrieved and archived the file and then checked for the next CDR file to retrieve and archive. Finding no additional CDR files, the last log message states there is no past CDR file.

Log Message (disable CDR collection)

```
Turning CDRCollection Off, there are <102> transactions in progress
for <osp>
Turned CDRCollection Off for <osp>
CDR collection disabled by user "UserName"
```

Explanation

`"UserName"` has disabled the CDR collection feature of the NexSRS server on the Features page, under the **Server** tab, of the user interface. The NexSRS server will reject any OSP *UsageIndication* messages it receives and will log the following message.

```
(Error 503 Service Not Available) UsageInd message rejected; Collect
CDRs feature is disabled, from Device <DeviceName>
```

License

License

Log Message (update License)

```
Created Configuration Back Up file <nexus_cfg_2002-12-27-13-52-07.bu>
Software license updated by user "UserName"
Validating license.
Updating license.
License updated.
```

Explanation

A new commercial software license for the NexSRS server was successfully installed using the Upgrade with new License Key link on the License page, under the **Server** tab, of the NexSRS server user interface.

License

Log Message (wrong IP address)
`Created Configuration Back Up file <nexus_cfg_2002-12-27-17-27-38.bu>`
`Software license updated by user "UserName"`
`Validating license.`
`Failed to validate IP Address`
`Failed to validate the license, license update canceled.`
`Error updating license`
 	
Reason for log Message
The IP address in the license does not match the IP address of the NexSRS server.
 	
Remedy
Upgrade with a software license that includes the IP address of the NexSRS server or change the IP address of the NexSRS server to match the IP address of the license.

Log Message (License expired)
`Created Configuration Back Up file <nexus_cfg_2002-12-27-18-43-50.bu>`
`Software license updated by user "UserName"`
`Validating license.`
`Failed to validate NotAfterDate`
`Failed to validate the license, license update canceled.`
`Error updating license`
 	
Reason for log Message
The license has expired. If the current date is later than the NotAfterDate specified in the NexSRS server license, then the license has expired.
 	
Remedy
Install a new license with a NotAfterDate later than the current date.

Log Message (version not licensed)
`Created Configuration Back Up file <nexus_cfg_2002-12-27-18-47-33.bu>`
`Software license updated by user "UserName"`
`Validating license.`
`Failed to validate NotAfterVersion`
`Failed to validate the license, license update canceled.`
`Error updating license`
 	
Reason for log Message
The current version of NexSRS server installed is newer than the NotAfterVersion value specified in the license.
 	
Remedy
Obtain a new NexSRS server license with a NotAfterVersion value greater than the current version of NexSRS server software, or install an earlier version of the NexSRS server.

License

Log Message (cut and paste error)
`Created Configuration Back Up file <nexus_cfg_2002-12-30-16-42-29.bu>`
`Software license updated by user "UserName"`
`Validating license.`
`Failed to extract Signature element`
`Failed to validate the license, license update canceled.`
`Error updating license`
 	
Reason for log Message
The complete license was not cut and pasted properly into the **Enter New License Key:** window of the NexSRS server user interface.
 	
Remedy
Click on the Upgrade with new License Key link on the License page under the **Server** tab of the NexSRS server user interface. The **Enter New License Key:** window will appear. Carefully cut and paste the entire license into the **Enter New License Key:** window.

Log Message (tampered licensed)
`Created Configuration Back Up file <nexus_cfg_2002-12-30-16-43-44.bu>`
`Software license updated by user -`
`Validating license.`
`Failed to validate the signature`
`Failed to validate the license, license update canceled.`
`Error updating license`
 	
Reason for log Message
The license is not valid because it has been changed after being signed.
 	
Remedy
Obtain a new license from support@transnexus.com.

License

Log Message (Failed to Validate IP Address)

```
Software license updated by user ospadmin
Validating license.
Failed to validate IP Address
License will not be updated
```

If a license that is 100% the correct IP address fails to validate with the log messages shown above, make certain the `/etc/nodename` file exists that matches the hostname within the `/etc/hosts` file. Below are the UNIX command line results for a properly configured NexSRS host computer.

```
$ cat /etc/hosts
10.0.6.41       NexSRS1
192.168.150.254 GKR01ATL
192.168.150.246 GKR02ATL
127.0.0.1       localhost

$ cat /etc/nodename
NexSRS1
```

If the file `/etc/nodename` does not exist you will get the following response when you issue the hostname command.

```
# hostname
unknown
```

For a properly configured host you should see something like the following.

```
$ hostname
NexSRS1
```

400 Error Code – Bad Request

400 Error Codes – Bad Request

Log Message (parsing error)

```
(Error 400 Bad Request) AuthRequest denied, failed to parse the XML
message from Device <DeviceName>
Failed to extract the <___> tag.
```

Reason for Log Message

The NexSRS Server failed to extract one or more of the required fields (<___> tag) from the AuthorizationRequest from Device `<DeviceName>`. Required fields are Timestamp, CallID, SourceInfo, DestinationInfo, Service and MaximumDestinations.

Remedy

If bad requests are logged only from specific devices, the problem is probably isolated to the client devices. If bad requests are logged from messages from all devices it may be a problem with the NexSRS server or with the network. One process to debug this problem is to analyze the wire traffic to and from the NexSRS server. OSP XML messages can be captured off the wire using an application such as snoop.

400 Error Codes – Bad Request
Log Message (enrollment failed) `(Error 400 Bad Request) Not supported message from Device <DeviceName>` **Reason for log message** An enrollment request from Device `<DeviceName>` using SCEP enrollment protocol was incorrectly addressed to the NexSRS server enrollment url followed by /OSP. **Remedy** In the OSP client (Device `<DeviceName>`), change the address of the NexSRS server enrollment url to end with /SCEP instead of /OSP.
Log Message (enrollment failed) `(Error 400 Bad Request) Not supported message from Device <DeviceName>` **Reason for log message** This is a valid XML formatted message, but the message is not supported. In this case, an enrollment request from Device `<DeviceName>` using TEP enrollment protocol was incorrectly addressed to the NexSRS server enrollment url followed by /OSP. **Remedy** In the OSP client (Device `<DeviceName>`), change the address of the NexSRS server enrollment url to end with /TEP instead of /OSP.

401 Error Code – Not Authorized

401 Error Codes – Not Authorized
Log Message (undefined reporting device) `(Error 401 Not Authorized) IP Address <-.-.-.-> is not in the routing table.` **Reason for log message** The IP address (`<-.-.-.->`) of the reporting device which sent the OSP *AuthorizationRequest* or *UsageIndication* is not defined as a device in the NexSRS Server. **Remedy** If the device sending the OSP *AuthorizationRequest* or *UsageIndication* (with the IP address `<-.-.-.->` shown in the log message) is a valid device, add the device to the NexSRS Server routing table configuration.
Log Message (device not enrolled) `(Error 401 Not Authorized) Device <DeviceName> is not enrolled.` **Reason for log message** The `<DeviceName>` device sending the OSP *AuthorizationRequest* has not enrolled with the NexSRS Server.

401 Error Codes – Not Authorized

Remedy
Enroll the device with the NexSRS Server.

Log Message (undefined transport device)
```
(Error 401 Not Authorized) SourceAlternate of type 'transport'
<DeviceName> is not in the routing table, from Device <DeviceName>
```

Reason for log message
The device defined in the OSP *AuthorzationRequest* by <SourceAlternate type="transport"> is not defined in the NexSRS server routing table.

Remedy
Add the SourceAlternate of type 'transport' <DeviceName> Device to the NexSRS server routing table configuration.

Log Message
```
(Error 401 Not Authorized) SourceAlternate of type 'h323' <DeviceName>
is not in the routing table, from Device <DeviceName>
```

Reason for log message
The device defined in the OSP AuthorzationRequest by <SourceAlternate type="h323"> is not defined in the NexSRS server routing table. The NexSRS server always authenticates <SourceAlternate type="transport"> devices. The NexSRS server will also authenticate <SourceAlternate type="h323"> devices when the AuthenticateSourceAlternateH323 parameter is set to Yes in the `proxy_config.sh` file of the NexSRS server.

Remedy
Possible remedies:

1. Add the SourceAlternate of type 'h323' <DeviceName> Device to the NexSRS server routing table configuration.
2. Edit the `proxy_config.sh` file so AuthenticateSourceAlternateH323=No. This will disable authentication of <SourceAlternate type="h323"> devices.

Log Message (reporting device may NOT originate)
```
(Error 401 Not Authorized) AuthReq to <"destination number"> is
denied.
Customer <"Customer Name"> of reporting device <DeviceName> may NOT
originate calls.
```

Reason for log message
The reporting device, the device which sent the OSP *AuthorizationRequest*, belongs to a Customer that may NOT originate calls.

Remedy
Go to the Customers page of the NexSRS server user interface and change the status of the <"Customer Name"> to May Originate.

401 Error Codes – Not Authorized

Log Message (transport device may NOT originate)
`(Error 401 Not Authorized) AuthReq to <"destination number"> is denied.`
`Customer <"Customer Name"> of SourceAlternate/transport device <DeviceName> may NOT originate calls.`
 	
Reason for log message
The proxy device for a VoIP call (defined by the SourceAlternate type='transport' field in the OSP AuthorizationRequest) belongs to a Customer that may NOT originate calls.
 	
Remedy
Go to the Customers page of the NexSRS server user interface and change the status of the <"Customer Name"> Customer to May Originate.

Log Message (h323 device may NOT originate)
`(Error 401 Not Authorized) AuthReq to <"destination number"> is denied.`
`Customer <"Customer Name"> of SourceAlternate/h323 device <DeviceName> may NOT originate calls.`
 	
Reason for log message
The source device (defined by the SourceAlternate type='h323', of the DeviceInfo fields in the OSP AuthorizationRequest) belongs to a Customer that may NOT originate calls.
 	
Remedy
Go to the Customers page of the NexSRS server user interface and change the status of the <"Customer Name"> Customer to May Originate.

404 Error Codes – Route Not Found

404 Error Codes – Route Not Found
Log Message (no route) `(Error 404 Not Found) Unable to find a dest route for number <____>, for reporting/transport/h323 devices <-.-.-.->/<-.-.-.->/<-.-.-.->.` **Reason for log message** No route was found to the destination number. The NexSRS server first tries to find a route based first on the Customer of the source device (<SourceAlternate type=h323>, or DeviceInfo device defined in the OSP AuthorizationRequest). If no route is available, or if the device is not defined, the NexSRS server next tries to find a route based on the Customer of the proxy device (<SourceAlternate type=transport> or DeviceInfo device). If the proxy device is not defined, the NexSRS server searches for a route based on the Customer of the reporting device which sent the OSP AuthorizationRequest. (In all known implementations, the reporting device is the same as the 'transport' device.)

404 Error Codes – Route Not Found
Remedy Add a route for at least one of the reporting/transport/h323 devices to the destination number.

503 Error Codes – Service Not Available

503 Error Codes – Service Not Available
Log Message (SSL not allowed) `(Error 503 Service Not Available) SSL is disabled, OSP requests are forbidden on the SSL port at this time.` **Reason for log message** A client device has attempted to send an OSP *AuthorizationRequest* or *UsageIndication* message to the SSL port 1443 of the NexSRS Server when the NexSRS server is configured with the SSL feature turned-off. **Remedy** If SSL is the desired configuration, go to the features page of the NexSRS Server user interface and turn-on SSL at the NexSRS Server. If SSL is not the desired configuration, re-configure the OSP client to communicate with the NexSRS server using http messages on port 1080 of the NexSRS Server.
Log Message (SSL required) `(Error 503 Service Not Available) SSL is enabled, OSP requests are forbidden on the non SSL port at this time.` **Reason for log message** A client device has attempted send an OSP *AuthorizationRequest* or *UsageIndication* message to the non SSL port 1080 of the NexSRS Server when the NexSRS server is configured with the SSL feature turned-on. **Remedy** If SSL is not the desired configuration, go to the features page of the NexSRS Server user interface and turn-off SSL at the NexSRS Server. If SSL is the desired configuration, re-configure the OSP client to communicate with the NexSRS server using https messages on port 1443 of the NexSRS Server.

503 Error Codes – Service Not Available
Log Message (routing turned-off) `(Error 503 Service Not Available) AuthRequest denied, feature is disabled, from Device <-.-.-.->` **Reason for log message** Routing feature of the NexSRS server has been disabled. **Remedy** Go the features page of the NexSRS server user interface and turn-on the Routing Authorization feature.
Log Message (BHCA exceeded) `(Error 503 Service Not Available) AuthRequest denied, BHCA has exceeded the limit, from Device <DeviceName>` **Reason for log message** The number of Busy Hour Call Attempts (BHCA) during the previous hour has exceeded the number of call attempts allowed by the NexSRS server's software license. If the number of call attempts per hour exceeds the licensed limit, the NexSRS server will refuse additional call attempts until the number of call attempts for the previous hour falls below the licensed number of BHCA. **Remedy** There are three possible remedies. 1. Wait for the average number of call attempts during the previous hour to fall below the licensed number of BHCA. 2. Install a new license with a greater licensed number of BHCA. 3. Direct OSP client devices to send OSP *AuthorizationRequests* to other NexSRS servers.
Log Message (CDR collection turned-off) `(Error 503 Service Not Available) UsageInd message rejected; Collect CDRs feature is disabled, from Device <DeviceName>` **Reason for log message** The Call Detail Collection feature of the NexSRS server has been disabled. **Remedy** Turn-on the Call Detail Collection feature of the NexSRS server. Call Detail Collection is enabled on the Features page, under the **Server** tab, of the NexSRS server user interface.

OSP Client Errors

Miscellaneous OSP Client Errors

Miscellaneous OSP Client Errors

Log Message (CDR rejected)

```
Failed to extract element <____>
UsageInd element rejected, failed to parse it, from Device
<DeviceName>
```

Reason for log message

An OSP UsageIndication message from Device <DeviceName> was rejected because it did not contain one or more of the following required fields: Timestamp, Role, TransactionID, CallID, SourceInfo, DestinationInfo, DestinationAlternate type=transport.

Remedy

If the problem is intermittent with OSP client devices, the problem may be a network problem. If the problem occurs only with certain OSP clients, it may be an OSP client specific problem.

Log Message (wrong url for NexSRS server)

```
Failed to parse the query string
```

Reason for log message

An OSP message was sent to NexSRS server url ending in \SCEP, not \OSP.

Remedy

In the OSP client device, edit the url of the NexSRS server to end with \OSP, not \SCEP.

Miscellaneous OSP Client Errors

Log Message (wrong url for NexSRS server)

```
Failed to parse the query string
<?xml version="1.0"?>
<Message messageId="242101" random="24210">
<AuthorizationRequest componentId="242100">
<Timestamp>2002-12-30T16:32:27Z</Timestamp>
<CallId encoding="base64">MQ==</CallId>
<SourceInfo type="e164">16789476602</SourceInfo>
<SourceAlternate type="transport">[1.1.1.1]</SourceAlternate>
<SourceAlternate type="subscriber">JohnDoe</SourceAlternate>
<SourceAlternate type="h323">[2.2.2.2]</SourceAlternate>
<DestinationInfo type="e164">16789476602</DestinationInfo>
<Service/>
<MaximumDestinations>8</MaximumDestinations>
<transnexus.com:CustomerId>
critical="False">1000</transnexus.com:CustomerId>
<transnexus.com:DeviceId
critical="False">1000</transnexus.com:DeviceId>
</AuthorizationRequest>
</Message>
```

Reason for log message
An OSP message was sent to NexSRS server url ending in \TEP, not \OSP.

Remedy
In the OSP client device, edit the url of the NexSRS server to end with \OSP, not \TEP.

Log Message (wrong url for NexSRS server)

```
Unrecognized script: /cgi-bin
```

Reason for log message
An OSP message was sent to NexSRS server url ending in \cgi-bin, not \OSP.

Remedy
In the OSP client device, edit the url of the NexSRS server to end with \OSP, not \cgi-bin.

ExtraNet

The following log messages are example log messages which will be seen on the Enterprise NexSRS Server. These log messages will not be seen on the Carrier NexSRS server.

ExtraNet

Log Message (Invalid IP address for Carrier NexSRS server)

```
ToolKit 14030: Host not found. Invalid address: <IP address or domain
name of Carrier NexSRS server>
```

Reason for log messages
The Carrier NexSRS server IP address or domain name is invalid. For example, if the IP address in the Inter Domain Carrier field of the license key is http:999.0.0.01080/osp,

ExtraNet

this message will occur because 999.0.0.0 is an invalid IP address.

Remedy
Make certain the IP addresses of the Carrier NexSRS servers in the Inter Domain Carrier field of license key are correct.

Log Message (401 Unauthorized)

```
ToolKit 14296: HTTP Status: 401 Unauthorized, IP Address: <IP address
of Carrier NexSRS server>
(Error 404 Not Found) Unable to find a dest route for number
<*01*250>, for reporting/transport/h323 devices
<172.16.4.8>/<8.7.2.6>/<3.2.3.0>,
```

Reason for log message
The Enterprise NexSRS Server is:

a) Not defined in the Carrier NexSRS server.
b) Is defined in Carrier NexSRS server, but does not have "enrolled" status.
c) Is defined in Carrier NexSRS server with "enrolled" status, but may not originate
d) Enterprise Server device and status are correctly configured in the Carrier NexSRS server, but `AUTHENTICATE_ACTUAL_SOURCE_DEVICE=YES` in the Carrier NexSRS server configuration and the source gateway is not defined in the Carrier NexSRS server.

Note that the log message 14296 may occur in pairs. This occurs because the Enterprise NexSRS server sends an OSP *AuthorizationRequest* to the first Carrier NexSRS server. The *AuthorizationRequest* fails and the first 14296 messages is written. The Enterprise NexSRS server resends the *AuthorizationRequest* to the second Carrier NexSRS server. It also fails and the second 14296 log message is written.

Remedy

1. Make certain the IP addresses of the Carrier NexSRS servers in the Inter Domain Carrier field of the Enterprise NexSRS server software license are correct.
2. If the Enterprise NexSRS server is behind a Network Address Translation device, such as a firewall or a load balancer, make certain the `SERVER_PUBLIC_IP_ADDRESS` parameter in the `inter_osp_config.sh` has been changed to the public IP address of the Enterprise NexSRS server as seen by external networks.
3. Contact the operator of the Carrier NexSRS server and make certain the public IP address of the Enterprise NexSRS server has been defined in the Carrier NexSRS server and that its status is set to "enrolled" and "may originate".
4. If `AUTHENTICATE_ACTUAL_SOURCE_DEVICE=YES` in the Carrier NexSRS server configuration, ask the Carrier NexSRS server operator to add the source gateway as a device to the Carrier NexSRS server routing table.

Log Message (Service Unavailable)

ExtraNet

```
ToolKit 14297: HTTP Status: 503 Service Unavailable, IP address: <IP
address or domain name of Carrier NexSRS server>
```

Reason for log messages
The Carrier NexSRS server is running and available, but routing and authorization have been turned off.

Remedy
Routing and authorization feature of Carrier NexSRS server must be enabled.

Log Message (Wrong URL for Carrier NexSRS server)

```
ToolKit 14298: HTTP Status: 400 Bad Request, IP address: <IP address
of Carrier NexSRS server>
ToolKit 14298: http response unexpected: Server unavailable at the URL
specified. Modify the URL and try again.
```

Reason for log messages
The Carrier NexSRS server URL is incorrect. For example, if the URL in the Inter Domain Carrier field of the license key is http:1.2.3.4:1080/tep, this message will occur because the URL needs to end with /osp.

Remedy
Make certain the URLs of the Carrier NexSRS servers in the Inter Domain Carrier field of license key are correct.

Log Message (IP Connection problems)

```
ToolKit 14300: Application slow in responding, response timed out to
IP address: <IP address of Carrier NexSRS server>
ToolKit 14310: Connection to application has been reset. IP address:
<IP address of Carrier NexSRS server>
ToolKit 14350: Cannot communicate with application. Connection timed
out to IP address <IP address of NexSRS server>
```

Reason for log messages
The three log messages above indicate TCP-IP connectivity problems between the Enterprise and Carrier NexSRS servers. These problems may be caused by network problems or because either the Enterprise NexSRS Server or Carrier NexSRS server is too busy to respond before the connection timed out.

Remedy
Occasional connection timeouts are normal. However, if connection timeouts exceed 2% of all ExtraNet calls during periods of heavy traffic, the problem should be investigated. If the problem is suspected to be an overloaded Enterprise NexSRS server, monitor CPU usage. If CPU usage is below 80% when the log messages occur, the connection time outs are probably not due to the Enterprise NexSRS server. If network problems are suspected, ping the Carrier NesSRS server. If ping times are less than 200 ms with no dropped packets, then network problems are probably not a major factor causing the connection timeouts. Report the problem to the Carrier NexSRS server

ExtraNet
operator.
Log Message (No Route) `ToolKit 11503: OSP Response Status: 404 Route Not Found (Error 404 Not Found) Unable to find a dest route for number <*01*250>, for reporting/transport/h323 devices <172.16.4.8>/<8.7.2.6>/<3.2.3.0>,` **Reason for log message** The Carrier NexSRS server does not have a route to terminate the call requested by the Enterprise NexSRS server. **Remedy** Carrier must add a destination route for the called number.

Questions and Answers

How to Restore a nexus_cfg.bu file

Q. I have not backed-up the nexus.cfg configuration file using the Back-up configuration to local file link on the Backup page, under the **Server** tab, on the user interface. I want to restore a previous nexus.cfg configuration. What can be done?

A. The NexSRS Server saves a back-up copy of the nexus.cfg configuration file to the $NexSRS_HOME/BACKUP directory at the following times:

- Right after start-up.
- Just before shutdown.
- Before the commercial license is updated.
- Before a configuration file is restored from the user interface.

Follow the procedures below to restore the desired nexus.cfg backup file.

1. Within the $NexSRS_HOME/BACKUP directory, search for the file you wish to restore. Example UNIX command and response are shown below.

```
$ls
nexus_cfg_2001-11-05-21-42-09.bu
nexus_cfg_2001-11-05-21-42-13.bu
nexus_cfg_2001-11-05-21-43-35.bu
nexus_cfg_2001-11-05-21-43-39.bu
nexus_cfg_2001-11-05-21-45-10.bu
```

2. Become superuser.

```
$su
Password:
```

3. Stop the NexSRS server. Search for the watchdog Process ID (PID) and kill the watchdog PID. Example UNIX commands are shown below.

```
# ps -ef | grep nexus
```

```
    root 14281 14280  0 21:32:45 pts/4    0:00 grep nexus
    root 14255 14254  0 21:25:39 pts/4    0:02 nexus_server
    root 14254     1  0 21:25:37 pts/4    0:00 ./nexus_watch_dog
# kill 14254
```

4. Make certain xitami, nexus_server, and watchdog processes have all stopped.

```
# ps -ef | grep nexus
    root 14284 14280  0 21:33:01 pts/4    0:00 grep nexus
# ps -ef | grep xitami
    root 14286 14280  0 21:33:05 pts/4    0:00 grep xitami
```

5. Copy the desired nexus.cfg backup file and replace nexus.cfg. In this example, the file is `nexus_cfg_2001-11-05-21-45-10.bu`.

```
# cp nexus_cfg_2001-11-05-21-45-10.bu nexus.cfg
```

6. Restart the server:

```
# ./start_osp_server.sh
Starting NexSRS Server with ENGINE_ID_NUM = 100 …
# exit
$
```

How to check CPU and Memory Usage

Q. How do I check CPU and memory usage of the Solaris machine running the NexSRS server?

A. Use the Solaris command "top" to obtain this information.

```
top -s1
```

Type "q" to quit top. An example of top is given below:

```
64 processes:  62 sleeping, 1 zombie, 1 on cpu
CPU states: 99.8% idle,  0.0% user,  0.2% kernel,  0.0% iowait,  0.0%
swap
Memory: 512M real, 157M free, 200M swap in use, 1142M swap free

   PID USERNAME THR PRI NICE  SIZE   RES STATE    TIME    CPU COMMAND
 20699 txusr      1  50    0 1752K 1320K cpu      0:00  0.95% top
 20644 txusr      1  59    0 7392K 4968K sleep    0:04  0.09% dtterm
 20646 txusr      1  48    0 1888K 1184K sleep    0:00  0.04% ksh
  1052 root      14  58    0   25M   23M sleep   62:32  0.03% nexus_server.ch
 20621 txusr      8  59    0 9672K 7632K sleep    0:03  0.02% dtwm
   422 root      12  58    0 5824K 3328K sleep   34:08  0.01% mibiisa
   432 oracle     1  59    0  149M  118M sleep   57:52  0.00% oracle
   185 root       5   8    0 4328K 3200K sleep   50:00  0.00% automountd
   434 oracle     1  59    0  148M  118M sleep   10:57  0.00% oracle
   439 oracle     1  58    0  148M  122M sleep    2:52  0.00% oracle
   437 oracle     1  59    0  147M  117M sleep    1:39  0.00% oracle
  1055 root       1  59    0 4224K 3304K sleep    1:25  0.00% xitami
   428 oracle     1  59    0  148M  116M sleep    0:35  0.00% oracle
 29988 root       1  58    0 2928K 1688K sleep    0:28  0.00% sendmail
```

```
   292 root       1  59    0 1016K  608K sleep    0:21  0.00% utmpd
```

If top cannot be found, look for it in the following directories:

```
/usr/local/bin, /usr/local/sbin, /usr/bin
```

Top can also be found by executing the following search command:

```
find / -name top 2>/dev/null
```

How to Check Disk Space

Q. How to check a Solaris machine for the amount of free space on the disk partition containing the NexSRS server?

A. This can be done by executing the "df" command:

```
df -k > diskspace.txt
cat diskspace.txt
```

An example of the output from these commands is given below:

```
Filesystem            kbytes    used   avail capacity  Mounted on
/dev/dsk/c0t0d0s0     480815  214786  217948    50%    /
/dev/dsk/c0t0d0s3    3007086 1961918  985027    67%    /usr
/proc                      0       0       0     0%    /proc
fd                         0       0       0     0%    /dev/fd
mnttab                     0       0       0     0%
/etc/mnttab
swap                 1167824       0 1167824     0%    /var/run
swap                 1168552     728 1167824     1%    /tmp
/dev/dsk/c0t0d0s5    6046389 1924504 4061422    33%    /app
/dev/dsk/c0t0d0s4    4030518 1231369 2758844    31%    /opt
/dev/dsk/c0t0d0s7    2627166 2530651   43972    99%    /oradbf1
/dev/dsk/c0t0d0s6    2505902  958630 1497154    40%    /usr/local
txsvr:/home          8255565 7701301  471709    95%
/txsvr/home
txsvr:/home1        17413250 15928515 1310603  93%
/txsvr/home1
```

How to Free-up Disk Space

Q. The NexSRS Server has run out of disk space and stopped. How do I free-up disk space?

A. The following procedure is one solution for quickly making more disk space available.

a) From the $NexSRS server directory, become superuser.

```
su
password: *******
```

b) Look at the size of the cdrs:

```
cd xitami/osp_ui/cdrs/ARCHIVE
ls -l
```

c) If there are lots of uncompressed files then compress the file using either of the following commands.

```
gzip *.cdr  or compress *.cdr
```

d) Go back to the $NexSRS_HOME directory

```
cd ../../..
```

e) Go to the log directory

```
cd xitami/osp_ui/log
```

f) Make a backup copy of the log file

```
mv nexus.log nexus.log.bak
```

g) Compress the logfile by executing either

```
gzip nexus.log.bak or compress nexus.log.bak
```

h) Go back to the $NexSRS_HOME directory and exit out of superuser

```
cd ../../..
exit
```

How to set NexSRS server time to UTC.

Q. How do I change the time clock of my NexSRS Server, running on a Solaris server, to UTC (Universal Time Coordinates, also know as GMT)?

A. To change the system time on the server, please become super-user and edit the /etc/default/init file and change TZ=GMT. Below is an example from Solaris.

```
# vi /etc/default/init
"/etc/default/init" [Read only] 11 lines, 455 characters
# @(#)init.dfl 1.2 92/11/26
#
# This file is /etc/default/init.  /etc/TIMEZONE is a symlink to this
file.
# This file looks like a shell script, but it is not.  To maintain
# compatibility with old versions of /etc/TIMEZONE, some shell constructs
# (i.e., export commands) are allowed in this file, but are ignored.
#
# Lines of this file should be of the form VAR=value, where VAR is one of
# TZ, LANG, or any of the LC_* environment variables.
#
TZ=GMT
```

Once this file has been edited, reboot the Solaris server.

CDRs lost from power failure

Q. If the power to the NexSRS Server fails and the machine crashes, will any CDRs be lost?

A. The NexSRS server writes CDRs to the current nexus cdr file every minute, or after 100 cdrs have accumulated, which ever come first. If the NexSRS server crashes, accumulated cdrs not written to file will be lost.

Corrupted nexus.cfg file

Q. What happens if a corrupted nexus.cfg file gets pushed to the NexSRS server?

A. The NexSRS server checks a new nexus.cfg file for syntax errors before updating its configuration. If a nexus.cfg file has errors, it will be rejected and the nexSRS server will continue to operate with its existing configuration.

License rejected because of invalid IP address.

Q. The NexSRS log message says that the license was rejected because the licensed IP address is invalid.

A. The IP address in the NexSRS license must be the primary IP address of the Sun Solaris server. Use the `ifconfig` command to determine the primary IP address of your server. The primary IP address is the `eri0` address as shown in the example below.

```
# ifconfig -a
lo0: flags=1000849<UP,LOOPBACK,RUNNING,MULTICAST,IPv4> mtu 8232 index 1
        inet 127.0.0.1 netmask ff000000
eri0: flags=1000843<UP,BROADCAST,RUNNING,MULTICAST,IPv4> mtu 1500 index 2
        inet 167.160.231.210 netmask fffffff0 broadcast 203.160.231.223
        ether 0:3:ba:10:13:fa
hme0: flags=1000843<UP,BROADCAST,RUNNING,MULTICAST,IPv4> mtu 1500 index 3
        inet 222.168.20.60 netmask ffffff00 broadcast 192.168.20.255
        ether 0:3:ba:1:51:a9
```

Next, make certain that your /etc/hosts file agrees with `ifconfig`. If not, correct the /etc/hosts file so it agrees with ifconfig as shown in the example below.

```
# cat /etc/hosts
#
# Internet hosts table
#
127.0.0.1 localhost
167.160.231.210 SOSS.VoIPCarrier loghost
222.168.20.60 SOSS1.VoIPCarrier
```

Chapter 11

Cisco Notes

This chapter contains supplemental information about operating the NexSRS peering server with Cisco VoIP gateways. The OSP peering protocol is supported on the 2600's, 3600's, 5300's, 5400's and 5800's VoIP gateways. To use OSP on these series of gateways requires IOS version 12.2(1), or later, and the "ENTERPRISE PLUS IPSEC 56" image or the "IP PLUS IPSEC 56" image.

Configuring Dial Peers

Once a Cisco gateway is enrolled with an NexSRS Server, the next step is to configure the gateway to use the NexSRS Server for settlement – or routing and call detail reporting. To do this the clock on the Cisco gateway must be synchronized with the same time used by the NexSRS Server and the dial-peers must to use application settlement. The configuration steps below are example of how a source and destination gateways can be configured for settlement.

Source gateway:

1. Take note of the physical port that the phone or T1 line is plugged into on the gateway.
2. Login to gateway
3. Type "en" along with the 'enable' password to go into 'enable mode'.
4. Set the time on the gateway:
    a. clock set hh:mm:ss dd Month yyyy
5. Type "config t" to go into configuration mode.
6. Set the timezone on the gateway:
    a. clock timezone EST 0 0
7. Configure an NTP server type:
    a. ntp server 129.79.5.100 (ip address can be different as long as it is for a valid NTP server)
8. To configure the incoming POTS peer type the following:
    a. dial-peer voice 1 pots (the tag "1" is arbitrary)

   b. application session
   c. incoming called-number 1404T (this peer will accept any digits starting with 1404)
   d. port 0:D (physical port where the T1 is connected on the PRI)
   e. exit

9. To configure the outgoing VoIP peer type the following:
   a. dial-peer voice 111 voip (the tag "111" is arbitrary)
   b. application session (tells the VoIP peer to use session target instead of the default application)
   c. destination-pattern 1404T (this peer will accept any digits starting with 1404)
   d. session target settlement:0 (connects the VoIP peer to the Settlement statement for the proper url for the NexSRS Server)
   e. exit
10. To configure the Settlement statement type the following:
   a. settlement 0
   b. type osp
   c. url http://172.16.4.25:1080/osp (this is the url of an
   d. NexSRS Server)
   e. no shutdown (enables Settlement)
   f. exit

Destination Gateway:

1. Take note of the physical port that the phone or T1 line is plugged into.
2. Login to gateway
3. Type "en" along with the 'enable' password to go into 'enable mode'.
4. Set the time on the gateway:
   a. clock set hh:mm:ss dd Month yyyy
5. Type "config t" to go into configuration mode.
6. Set the timezone on the gateway:
   a. clock timezone EST 0 0
7. Configure an NTP server type:
   a. ntp server 129.79.5.100 (ip address can be different as long as it is for a valid NTP server)
8. To configure the outgoing POTS peer type the following:
   a. dial-peer voice 1 pots (the tag "1" is arbitrary)
   b. application session
   c. destination-pattern 1404T (this peer will accept any digits starting with 1404)
   d. port 0:D (physical port where the T1 is connected on the PRI)

e. exit

9. To configure the incoming VoIP peer type the following:
    a. dial-peer voice 111 voip (the tag "111" is arbitrary)
    b. application session (tells the VoIP peer to use session target instead of the default application)
    c. incoming called-number 1404T (this peer will accept any digits starting with 1404)
    d. session target settlement:0 (connects the VoIP peer to the Settlement statement for the proper url of the NexSRS Server)
    e. exit
10. To configure the Settlement statement type the following:
    a. settlement 0
    b. type osp
    c. url http://172.16.4.25:1080/osp (this is the url of an NexSRS Server)
    d. no shutdown (enables Settlement)
    e. exit

Using Unsigned Tokens

The default settlement configuration for Cisco gateways expects OSP tokens to be signed. If you use unsigned tokens, configure your Cisco gateway to validate unsigned tokens with the following commands. Note, this setting is required when using the Linux version of NexSRS.

```
Router# conf t
Router(config)#service internal
Router(config)#settlement 0
Router(config-settlement)#signed-token-off
```

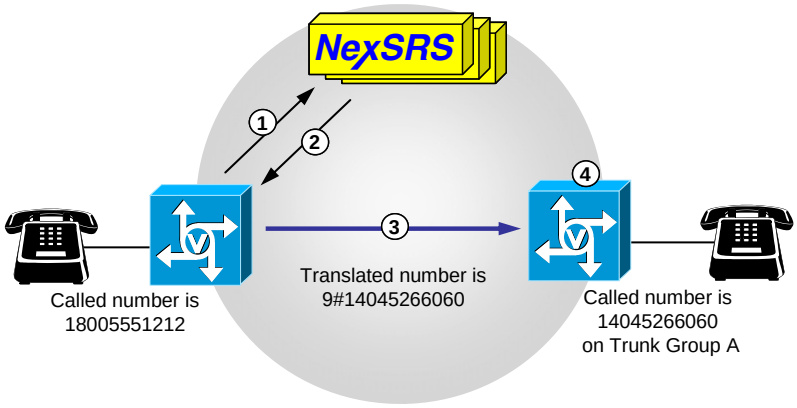
More information on Cisco Gateways

Note* Please reference the following Cisco document for a more detailed description of Dial-Peers:
http://www.cisco.com/univercd/cc/td/doc/product/software/ios122/122cgcr/fvvfax_c/vvfpeers.htm

Note 2 * Please reference the following Cisco document for a more detailed description of Settlement:
http://www.cisco.com/univercd/cc/td/doc/product/software/ios122/122cgcr/fvvfax_c/vvfosp.htm#12268

Trunk Group Routing

The NexSRS peering server supports native trunk group routing for Cisco H.323 gateways. However, trunk group routing can also be implemented by using number translation on the NexSRS peering server coupled with dial peer configuration on a Cisco gateway. This trunk group routing technique may be used with both H.323 or SIP gateways, but is most useful for SIP implementations which lack the trunk group features of H.323. The following call scenario provides an example:



1. The calling party dials 18005551212. The call is routed from the PSTN to a source gateway. The source gateway sends a peering request to the NexSRS peering server.
2. The NexSRS server is configured to translate the called number 18005551212 from the source peer to 9#14045266060. The NexSRS returns a peering response to the source gateway with the translated called number and the IP address of the destination gateway or peer.
3. The source peer routes the call to the destination peer with the translated called number 9#14045266060.
4. The destination Cisco gateway has been configured to remove the 9# prefix from the called number and route the VoIP call to the PSTN on Trunk Group A.

Cisco Gateway Configuration

Below is an example of the dial-peer configuration on the Cisco destination gateway which will remove the 9# prefix and route all calls to the Trunk Group A.

```
voice translation-rule 1
 rule 1 /^.*#\(.*\)/ /\1/
```

This entry instructs the Cisco gateway to remove the # and any number-prefix in front of the # from the called number.

```
dial-peer voice 100 pots
 description Trunk Group A
 translate-outgoing called 1
 destination-pattern 9#.T
 port 1:D
```

This pots dial-peer on the Cisco gateway defines the translation rule for Trunk Group A and provides the gateway with the following instruction:

- `translate-outgoing called 1` - use rule 1 defined above
- `destination-pattern 9#.T` - this dial-peer applies to any call with called number beginning with 9#
- `port 1:D` - route call to PSTN on physical trunk group 1:D

Suggested Routes

Cisco gateways may be configured to include the IP address of the destination gateway in the OSP *AuthorizationRequest* sent to the NexSRS Server. This feature, known as "suggested

routes", configures the source gateway to suggest the destination gateway to the NexSRS Server which then determines if the source gateway may access the destination gateway. If the routing policy in the NexSRS Server allows calls to be made from the source gateway to destination gateway, the NexSRS Server will return an OSP *AuthorizationResponse* with a digitally signed token authorizing the call. When suggested routes feature is used, the IP address of the suggested destination gateway is included in the <*DestinationAlternate type="transport"*> element of the OSP *AuthorizationRequest*.

Suggested routes are configured in Cisco gateways by entering the following dial-peer configuration:

```
dial-peer voice 999 voip
  destination pattern 408.......
  session target ipv4:1.2.3.4
  settle-call
```

In this example, destination pattern is the called number dial-string and sesion target is the IP address of the destination gateway. This feature has been implemented on Cisco 3600 and 5300 gateways running IOS 12.1(1)T or later.

Chapter 12

Appendices

1. Release Notes

NexSRS release numbers are defined by three numbers separated by periods, for example: X.Y.Z, where X defines a major release, Y defines a release with one or more new features and Z defines a new release with bug fixes only and no new features.

```
Version 3.0.4 / Build 2007-04-30 / Tag NS-3_0_4_20070430
  o Updated online User Manual

Version 3.0.3 / Build 2007-04-19 / Tag NS-3_0_3_20070419
  o Implemented Transaction ID Version 2
  o Added support for Java Applet Viewer 1.6.0

Version 3.0.2 / Build 2007-04-04 / Tag NS-3_0_2_20070404
  o BUG-FIX: in logic for searching devices by IP address and
    network id.

Version 3.0.1 / Build 2007-03-15 / Tag NS-3_0_1_20070315
  o Added 3 new CDR fields for reporting, source and destination device
    names.
  o Do not translate calling numbers containing non-routing characters.
  o Do not require full match by IP address and network id.
  o Limit size of the nexus.cdr file.
  o Added log messages for cases when a suggested destination is not
    authorized or a route request is blocked.

Version 3.0.0 / Build 2007-01-02 / Tag NS-3_0_0_20070102
  o Do not include OSP tokens for calls between Gateways and non-osp
    destinations
  o Added 'In Tech Prefix' to the devices-by-address primary key
  o Added a new device property - Strip in tech Prefix.
  o Regrouped device properties by identity, property and status
  o Regrouped html pages by route provisioning, number translation and
    server (split the 'backup and restore configuration page' into two
    - route provisioning and number translation)
  o Renamed route plans to dial plans
    (renamed route-plans-[1234].cfg to dial_plan-[1234].cfg)
  o Renamed generic customers to 'Products'
```

```
Version 2.9.1 / Build 2006-06-09 / Tag NS-2_9_1_20060609
  o Updated online user manual
  o Updated default password file
  o Updated default software license
  o Updated license page to include a registration link
  o Updated applets to work with the latest patches for IE 6.0
  o Updated code signing certificate for applets
  o Updated README.txt
  o Optimized inter-process messaging between the OSP and HTTP servers
  o Bug-fix: correctly store the called number after routing
    manipulation in the internal CDR
  o Bug-fix: correctly order transaction id bits on x86 platforms

Version 2.9.0 / Build 2006-05-04 / Tag NS-2_9_0_20060504
  o Replaced Groups with Real and Generic Customers
  o Implemented destinations and rank-ordered load balancing
  o Increased maximum length of customer and destination names to 30
    characters
  o Upgraded Java Applet Viewer from 1.4.0 to 1.5.x
  o Ported HTML, CSS, and applet pages to FireFox 1.5
  o Ported source code to Linux
  o Implemented protocol specific call ids
  o Temporarily dropped support for look-ahead routing
  o Do not check 'may originate' during the route plan assignment logic

Version 2.8.1 / Build 2006-02-13 / Tag NS-2_8_1_20060213
  o Upgraded the version of the OSP toolkit from 3.1.0 to 3.3.4
  o Fixed an interoperability issue in validating look-ahead enabled
    tokens using older OSP toolkits
  o Changed the default value for USE_DEFAULT_ROUTE_PLAN_ASSIGNMENTS in
    start_osp_server.sh from 'yes' to 'no'

Version 2.8.0 / Build 2006-01-05 / Tag NS-2_8_0_20060105
  o Added 4 sets of route plan assignments
  o Added pre-routing called number digit filtering
  o Added routing called number manipulation
  o Added a new CDR field - 'Called Number After Routing Number
    Manipulation'
  o Added trace log messages for monitoring routing decisions
  o All configuration files have been moved to the new 'etc' sub-folder
  o Dropped support for 'implicit proxy routing'
  o Dropped support for 'detecting loop-backs in customer-specific
    routing'
  o Bug-Fix: updating routes from the GUI was changing the status for
    all devices to 'on-line'

Version 2.7.0 / Build 2005-11-14 / Tag NS-2_7_0_20051114
  o Added interface to the NexENUM module
  o Updated on-line documentation
  o Updated the password file

Version 2.6.0 / Build 2005-10-19 / Tag NS-2_6_0_20051019
  o Added a new CDR field - call connect time
  o Added copy/paste/select-all options to the routes pages
```

```
Version 2.5.3 / Build 2005-05-27 / Tag NS-2_5_3_20050527
  o Ignore XML parsing errors when XML syntax validation is turned off

Version 2.5.2 / Build 2005-05-13 / Tag NS-2_5_2_20050513
  o Bug-fix: buffer overflow in signing tokens with look ahead routing

Version 2.5.1 / Build 2005-01-07 / Tag NS-2_5_1_20050107
  o Added a new CDR field for storing signaling IP address in internal
    CDRs
  o Bug-fix: stop_osp_server.sh did not work on Solaris 9
  o Bug-fix: options 2 and 3 did not work in
    CHECK_FOR_DUPLICATE_DESTINATIONS logic
  o Bug-fix: a race condition in XML parsing logic
  o Bug-fix: a typo in expected gric extension element for reporting
    TCCode

Version 2.5.0 / Build 2004-12-06 / Tag NS-2_5_0_20041206
  o Added new destination protocol - IAX
  o Added new token format for terminating devices supporting IAX or
    SIP protocols
  o Improved error reporting in XML parser

Version 2.4.0 / Build 2004-11-24 / Tag NS-2_4_0_20041124
  o Added new device property - network id
  o Upgraded XML parser
  o Changed default CDR_FILE_TIME_SPAN from 60 to 5 minutes
  o Bug-fixes:
  o Create backup copies of all number translation rules on license
    update
  o Block calls in post-routing translation by sending OSP response 200
  o Limit In/Out prefixes and Calling Number/Called Number/Translation
    Rule to 32 characters

Version 2.3.0 / Build 2004-11-03 / Tag NS-2_3_0_20041103
  o Added post-routing calling number translation
  o Added seven new CDR Fields:
    - Calling Number in routing authorization request
    - Calling Number after pre-routing calling number translation
    - End of call time
    - Alert time for call
    - Reported Post Dial Delay
    - Originating Network ID (Source trunk group)
    - Terminating Network ID (Destination trunk group)

Version 2.2.1 / Build 2004-10-08 / Tag NS-2_2_1_20041008
  o Updated on-line help manual
  o Improved user interface to work on localized versions of Windows.
  o Removed Basic Configuration instructions from xitami.aut.
  o Optimized server-side logic for viewing large configurations.
  o CDR changes -
    - For calls originated by subnets, store the actual address instead
      of the subnet mask.
    - Store call start time for all call attempts reported in source
```

```
      UsageIndication messages.
    - Added four new CDR fields
      o Called Number Authorization Request from source device
      o Called Number After authenticating and removing Inbound Prefix
      o Called Number After Pre-Routing Number Translation
      o Called Number After Post-Routing Number Translation

Version 2.2.0 / Build 2004-08-26 / Tag NS-2_2_0_20040826
  o Added subnets.
  o Added support for optional content compression during configuration
    replication.
  o Added a new script for compressing backed up configuration files.
  o Extended allowed range for the maximum authorized call length from
    24 hours to 90 days.
  o Implemented numerous usability improvements for the user interface.

Version 2.1.0 / Build 2004-06-30 / Tag NS-2_1_0_20040630
  o Added post-routing number translation.
  o Check CDR fields for non-ASCII and control chars and replace them
    with spaces.

Version 2.0.3 / Build 2004-05-12 / Tag NS-2_0_3 20040512
  o Extended number translation rules to support adding suffixes.
  o Added a new page to the user interface for viewing All Devices.
  o Added utility script, delete_archived_nexus_log_files.sh for
    deleting archived log files.
  o Added a short-cut feature for adding a range of devices with
    sequential IP addresses.
  o When adding a new destination, the device must be explicitily
    selected.
  o Configured equal load balancing for ExtraNet messages.  Message
    counts (used for reconnection) are set to 1,000 and 1,000.
  o Increased allowable range of device weights from 0-999 to 0-10,000.

Version 2.0.2 / Build 2004-04-20 / Tag NS-2_0_2 20040420
  o Updated on-line help
  o Updated and created new utility scripts in new /ospnexus/utils
    directory
     - compress_archived_cdr_files.sh
     - delete_old_cdr_files.sh
     - cut_nexus_log.sh
     - delete_old_backup_files.sh
  o Upgraded OSP Toolkit to version 3.1.0
  o Removed TransID_decode file
  o Fixed a potential race condition in Watch Dog

Version 2.0.1 / Build 2004-04-14 / Tag NS-2_0_1_20040414
  o Updated look and feel of the web-based user interface
  o Updated license agreement
  o Updated startup and configuration scripts
  o Updated authentication configuration file for Xitami
  o Upgraded OSP ToolKit to version 3.0.0
  o Renamed configuration variable AUTHENTICATE_SOURCE_ALTERNATE_H323
    to AUTHENTICATE_ACTUAL_SOURCE_DEVICE and moved it from
```

```
    proxy_config.sh to start_osp_server.sh
  o Renamed license name to "NexSRS_server"
  o Added copyright notice to the license web page
  o Added a link to view the software license agreement
  o Added a new page for downloading archived CDR files
  o Moved license from nexus.cfg to nexus.lic file
  o Introduced a two second delay between starting Xitami and OSP
    server
  o The CDR file time span is now configurable in the start script
  o Bug-fix: return destination properties to devices supporting OSP
    version 2.1.1 or 2.1.1-P

Version 2.0.0 / Build 2004-03-02 / Tag NS-2_0_0_20040302
  o Implemented LookAhead routing
  o Extended device attribute OSPVersion with 2.1.1-P to indicate proxy
    devices capable of handling LookAhead routes

Version 1.11.1 / Build 2004-02-15 / Tag NS-1_11_1_20040215
  o Added support for Device Info element in OSP messages.
  o Upgraded OSP client to version 2.10.1
  o Fixed a bug in number translation introduced in version 1.11.0

Version 1.11.0 / Build 2004-02-04 / Tag NS-1_11_0_20040204
  o Added new device property indicating if device is on- or off- line
  o Added overwrite protection to the number translation page
  o Added EngineId validation to the start_osp_server.sh script
  o Enabled user interface to update enrolled device status
  o Optimized server-side logic for handling configuration updates
  o Minor bug-fix in license update module

Version 1.11.1 / Build 2004-02-15 / Tag NS-1_11_1_20040215
  o Added support for Device Info element in OSP messages.
  o Upgraded OSP client to version 2.10.1
  o Fixed a bug in number translation module introduced in version
    1.11.0

Version 1.10.4 / Build 2004-01-14 / Tag NS-1_10_4_20040114
o If a CDR can not be written to disk, log an error message keep
  the record in memory and cease accepting AuthorizationRequest and
  CDRs until can be written to disk.
o Insure that an engine id has been configured before starting the
  server.
o Calling number is no longer required in an OSP message.
o Improved algorithm for generating transaction ids.
o Added methods for exporting configuration into XML format

Version 1.10.3 / Build 2003-11-03 / Tag NS-1_10_3_20031103
o Disabled crypto hardware support for license verification.

Version 1.10.2 / Build 2003-10-14 / Tag NS-1_10_2_20031014
o Serialized access to crypto and CDR files.

Version 1.10.1 / Build 2003-10-10 / Tag NS-1_10_1_20031010
o Generate OSP tokens compatible with older toolkit versions.
```

o Check disk space availability in CDR's folder.
o Added a configuration variable for relaxed XML validation.
o Added configuration overwrite check for requests submitted from the routes applet.
o New user name and password prompt responds to mouse events.
o Eliminated a potential client-side caching issue while backing up number translation config.
o Eliminated a potential race conditions in the tech prefix translation logic.
o Updated the demo license

Version 1.10.0 / Build 2003-09-03 / Tag NS-1_10_0_20030903
o Eliminated potential race conditions in licensing and shutting down logic.
o Combined nexus_server.cha_yes and nexus_server.cha_no binaries into a single executable. Cryptographic hardware support can be enabled by changing the ENABLE_CRYPTO_HW_ACCELERATION=no/yes defined in the start_osp_server.sh script.
o Using memory allocation package optimized for SMP.
o Disabled Nagel's algorithm; using internal buffering instead.
o Removed synchronization overhead in the web server.
o Tokens with exact authorization parameters are now cashed and reused.

Version 1.9.1 / Build 2003-08-23 / Tag NS-1_9_1_20030823
o Eliminated a potential race condition in load balancing module.

Version 1.9.0 / Build 2003-08-05 / Tag NS-1_9_0_20030805
o Replaced HTML routes page with an applet-based implementation.
o Included an optional 64-bit version of Xitami.

Version 1.8.0 / Build 2003-06-25 / Tag NS-1_8_0_20030625
o Added Time/Day of Week logic to number translation.

Version 1.7.1 / Build 2003-06-03 / Tag NS-1_7_1_20030603
o Added edit capabilities to the devices page.

Version 1.7.0 / Build 2003-05-19 / Tag NS-1_7_0_20030519
o Added five new device properties: Short Name, DestinationProtocol, OSP Version, and Inbound /Outbound Prefixes.
o Added additional authentication using optional Inbound prefix.
o Added additional number translation logic for removing optional inbound prefix and adding optional outbound prefix.
o The server now provides destination device OSP version and signaling protocol capabilities in AuthResponse to clients that support OSP version 2.1.1
o Replaced HTML devices page with an applet-based implementation. This version provides view-only functionality.
o Removed device information from the groups page.

Version 1.6.2 / Build 2003-05-13 / Tag NS-1_6_2_20030513
o The server no longer blocks pending transactions while its routing configuration is being updated.

Version 1.6.1 / Build 2003-05-06 / Tag NS-1_6_1_20030506
o Improved error checking while parsing OSP messages.
o De-serialized access to the XML module.

Version 1.6.0 / Build 2003-04-28 / Tag NS-1_6_0_20030428
o Moved back up files to BACKUP folder
o Upgraded Java applet viewer version to 1.4.1
o Added number translation module

Version 1.5.2 / Build 2003-03-07 / Tag NS-1_5_2_20030307
o Implemented partial call blocking.

Version 1.5.1 / Build 2003-02-25 / Tag NS-1_5_1_20030225
o Improved performance of View, Add and Edit route pages.
o Add device to a route action no longer forces the 1st device to have a non-zero weight.
o Added web user info to UI log messages for deleting routes and restoring configuration.
o Added version number to distribution name and installation directory.
o Changed port numbers from 80,81,82 and 443 to 1080,1081,1082 and 1443 respectively.
o Changed auto_start_stop_osp_server.sh script to start and stop OSP server as user ospadmin.
o Starting and stopping the server no longer requires root account.

2. Load Balancing Algorithm

The NexSRS Server responds to an OSP *AuthorizationRequest* by providing and an OSP *AuthorizationResponse* that includes a list of *DestinationSignalAddress*es (IP addresses of destination points). In simple terms, the list of destination IP addresses will be ordered first with destination devices with weights greater than one followed with a random order of back-up devices which have a weight of zero. This Appendix provides a detailed description of the load balancing algorithm which determines the order of destination IP addresses in an OSP *AuthorizationResponse*.

The NexSRS Server load balancing algorithm uses the following variables to determine the number of Primary and Back-up devices.

Input Variables	**Possible Values**	**Description**
W	Any integer 0 to 999	Weight assigned to destination IP address
P	Any integer 0 to 12	Number of Primary devices (devices with weight greater than 0) that may terminate calls.
B	Any integer 0 to 12	Number of Back-up devices (devices with weight = 0) that may terminate calls.
R	Any integer > 1	Number of destination devices requested by the source. (<MaximumDestinations> in *AuthorizationRequest*)

In general, the load balancing algorithm will return R-1 Primary devices (randomly load balanced according to their weights) and 1 (randomly chosen) Back-up device as the last choice. However, there are two special cases:

1. If there are no Back-up devices, R number of Primary devices will be returned (randomly load balanced according to their weights).
2. If there are less than R-1 Primary devices (P<R-1), return P Primary devices (randomly load balanced according to their weights) and R-P Back-up devices (randomly chosen) as the last devices in the list.

The following example illustrates the general case for load balancing.

R = 3, Source device requests 3 destination addresses
P = 3, Three Primary devices (weight > 0) are possible route destinations
Device 1 | Weight 999
Device 2 | Weight 1
Device 3 | Weight 2
B = 7, Seven Backup devices (weight = 0) are possible route destinations
Backup 1| Weight 0
Backup 2| Weight 0
Backup 3| Weight 0
Backup 4| Weight 0
Backup 5| Weight 0

Backup 6| Weight 0

Step 1. The algorithm will randomly selects a uniformly distributed variable between 1 and 1002 (999+2+1). For this example assume the random value is 999.5555. The algorithm then orders devices by load factor in decreasing order and determines ranges for randomly selecting the first choice as shown below. In this case Device 3 is the first choice.

Device	Weight	Range	Random Value
1	999	0 and <999	
3	2	999 and <1001	999.5555
2	1	1001 and <1002	

Step 2. The algorithm repeats the process and selects a uniformly distributed variable between 1 and 1000 (999+1) to determine the second choice device. For this example assume the second random variable is 32.8572 which means Device 1 is selected as the second choice.

Device	Weight	Range	Random Value
1	999	0 and <999	32.8572
2	1	999 and <1000	

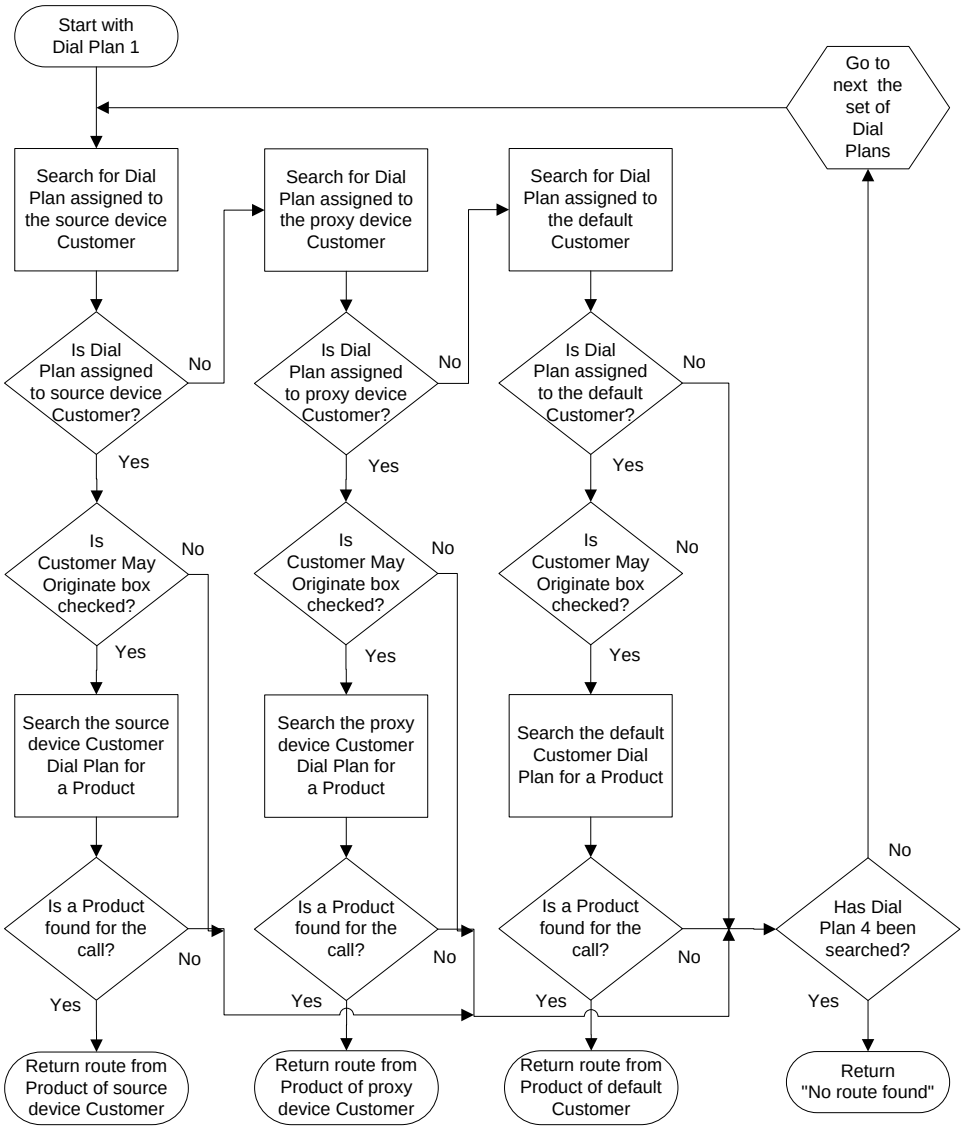
The algorithm has now selected R-1 Primary devices, where R = 3 is the total number of destinations requested by the source device. The 3rd Primary device will not be selected in order to accommodate a Backup device in the list of destination devices.

Step 3. The algorithm now randomly selects one backup device. For this example assume the randomly selected device is Backup 5. The final list of route destinations returned to the source device is:

1st Choice: Device 3
2nd Choice Device 1
3rd Choice Backup 5

3. Dial Plan Selection Logic

The flowchart below describes the Dial Plan selection algorithm. In this diagram a route found can be either a list of destination devices, or an explicitly blocked route.


Start with Dial Plan 1
Go to next the set of Dial Plans
Search for Dial Plan assigned to the source device Customer
Search for Dial Plan assigned to the proxy device Customer
Search for Dial Plan assigned to the default Customer
Is Dial Plan assigned to source device Customer?
Is Dial Plan assigned to proxy device Customer?
Is Dial Plan assigned to the default Customer?
No
Yes
Is Customer May Originate box checked?
Search the source device Customer Dial Plan for a Product
Search the proxy device Customer Dial Plan for a Product
Search the default Customer Dial Plan for a Product
Is a Product found for the call?
Has Dial Plan 4 been searched?
Return route from Product of source device Customer
Return route from Product of proxy device Customer
Return route from Product of default Customer
Return "No route found"

4. NexSRS Routing Table – nexus.cfg

Route Plans for the NexSRS server are stored in a text configuration file named nexus.cfg. Routing data in the nexus.cfg file is divided into fine sections: customers, destinations, devices, term-devices and routes. The following column definitions define the data fields for each section.

Customer Section

Column	Description	Acceptable entry
1	Data type	'customer'
2	Customer Name	Any combination of alphanumeric characters, the forward slash(/), underscore (_) and the dash (-). The Customer Name may be up to 25 characters long and may include single spaces between the characters.
3	May Originate status	'may_originate' or 'may_not_originate'
4	Customer Type	'real' or 'generic' – A generic customer is a Product

Destination Section

Column	Description	Acceptable entry
1	Data type	'destination'
2	Destination Name	Any combination of alphanumeric characters, the forward slash(/), underscore (_) and the dash (-). The Destination Name may be up to 25 characters long and may include single spaces between the characters.
3	Throttle	Any integer from 1 to 100
4	Increment	Any integer from 1 to 10

Devices Section

Column	Description	Acceptable entry
1	Data type	'device'
2	Device Address	IP address or any fully qualified domain name which can be resolved by a DNS server
3	Customer Name	The Customer to which the device is assigned. The Customer Name must exist in the nexus.cfg file and the Customer must be real.
4	May Terminate status	'may_terminate' or 'may_not_terminate'
5	Enroll status	'enrolled' or 'not_enrolled"
6	Short Name	Any combination of alphanumeric characters 0-9 and A-Z (upper and lower case), the forward slash(/), underscore (_) and the dash (-). The Short Name may be up to 25 characters long and may include single spaces between the characters. Short Name may be identical to Device Address.
7	Destination Protocol	'h323-Q931', 'h323-LRQ' or 'sip'
8	OSP Version	'0.0.0', '1.4.3' or '2.1.0'
9	Inbound Prefix	Numeric digits 0-9, comma (,), hash (#) or asterisk (*)
10	Outbound Prefix	Numeric digits 0-9, comma (,), hash (#) or asterisk (*)
11	Online status	'online' or 'offline'
12	Subnet Bit Mask	1-31 identifies a Subnet range. Bit Mask = 32 indicates a VoIP device
13	Network ID	Integers 0-9, Alpha A-Z, forward slash (/), underscore (_), dash (-).
14	Strip In Prefix	'TRUE' or 'FALSE', TRUE indicates the In Prefix should be stripped.

Term-devices Section

Column	Description	Acceptable entry
1	Data type	'term-devices'
2	Destination Name	Any Destination configured in the Destination section of nexus.cfg
4	Termination Device	Short Name of any device defined in nexus.cfg
5	Device Weight	Any integer from 1 to 10000
6	Termination Device	Short Name of any device defined in nexus.cfg
7	Device Weight	Any integer from 1 to 10000
8	Termination Device	Short Name of any device defined in nexus.cfg
9	Device Weight	Any integer from 1 to 10000

Note: There is no limit to the number of Termination Device and Weight pairs that may be configured for a route.

Routes Section

Column	Description	Acceptable entry
1	Data type	'route'
2	Customer Name	The Customer to which the route is assigned. The Customer Name must exist in the nexus.cfg file.
3	Calls To number	Numeric digits 0-9, comma (,), hash (#) or asterisk (*). Up to 32 digits
4	Destination	Destination defined in the Destination section of the nexus.cfg
5	Destination Rank	Any integer from 1 to 100
6	Destination Weight	Any integer from 1 to 10000
7	Destination	Destination defined in the Destination section of the nexus.cfg
8	Destination Rank	Any integer from 1 to 100
9	Destination Weight	Any integer from 1 to 10000

Note: There is no limit to the number of Destination, rank and weight triplets that may be configured for a route.

An example of a nexus.cfg file, imported to Microsoft Excel is provided on the following page.

Example of a nexus.cfg file imported into Excel

	A	B	C	D	E	F	G	H	I	J	K	L	M	N
3	Column 1	Column 2	Column 3	Column 4	Column 5	Column 6	Column 7	Column 8	Column 9	Column 10	Column 11	Column 12	Column 13	Column 14
4	routing	enabled												
5	cdrs	enabled												
6	customer	'default'	may_originate	real										
7	customer	'IP Telco'	may_originate	real										
8	customer	'ITSP'	may_not_originate	real										
9	customer	'IXC'	may_originate	real										
10	customer	'Low Cost Quality'	may_not_originate	generic										
11	customer	'Off-Peak'	may_not_originate	generic										
12	customer	'Peak'	may_not_originate	generic										
13	customer	'Premium Quality'	may_not_originate	generic										
14	customer	'PTT'	may_originate	real										
15	customer	'Suspended IXC Devices'	may_not_originate	real										
16	customer	'Telco North'	may_originate	real										
17	customer	'Telco South'	may_originate	real										
18	customer	'Week Day'	may_not_originate	generic										
19	customer	'Weekend'	may_not_originate	generic										
20	destination	'Hong Kong'	100	1										
21	destination	'IXC London'	100	1										
22	destination	'IXC Los Angeles'	100	1										
23	destination	'IXC New York'	100	1										
24	destination	'Mexico'	100	1										
25	destination	'PTT VoIP Operations'	100	1										
26	destination	'Telco North POP'	100	1										
27	destination	'Telco South NOC'	100	1										
28	device	'208.57.120.64'	'ITSP'	may_not_terminate	enrolled	'208.57.120.64'	h323-Q931	1.4.3	'1#'	''	on-line	32	''	TRUE
29	device	'208.57.120.65'	'ITSP'	may_terminate	enrolled	'Hong Kong GW.65	h323-Q931	1.4.3	'1#'	''	on-line	32	''	TRUE
30	device	'208.57.120.66'	'ITSP'	may_terminate	enrolled	'Hong Kong GW.66	h323-Q931	1.4.3	'1#'	''	on-line	32	''	TRUE
31	device	'208.57.120.67'	'ITSP'	may_terminate	enrolled	'Hong Kong GW.67	h323-Q931	1.4.3	'1#'	''	on-line	32	''	TRUE
32	device	'208.57.120.68'	'ITSP'	may_terminate	enrolled	'Hong Kong GW.68	h323-Q931	1.4.3	'1#'	''	on-line	32	''	TRUE
33	device	'208.57.120.69'	'ITSP'	may_terminate	enrolled	'Hong Kong GW.69	h323-Q931	1.4.3	'1#'	''	on-line	32	''	TRUE
34	device	'192.168.1.199'	'IXC'	may_terminate	enrolled	'192.168.1.199'	h323-Q931	1.4.3	''	''	on-line	32	''	TRUE
35	device	'189.45.72.20'	'IXC'	may_not_terminate	enrolled	'GW_A'	h323-Q931	1.4.3	''	''	off-line	32	''	TRUE
36	device	'189.22.72.0'	'IXC'	may_not_terminate	not_enrolle	'IXC_Subnet'	h323-Q931	1.4.3	''	''	off-line	30	''	FALSE
37	device	'128.47.88.6'	'PTT'	may_terminate	not_enrolle	'Gatekeeper'	h323-LRQ	2.1.1	'1#'	''	on-line	32	'TrunkGrp_A'	TRUE
38	device	'128.47.89.13'	'PTT'	may_terminate	enrolled	'SIP-H323 proxy'	sip	2.1.1-P	''	''	on-line	32	''	TRUE
39	device	'123.44.87.12'	'Suspended IXC Devices'	may_terminate	enrolled	'GW_B'	h323-Q931	1.4.3	''	'11'	off-line	32	''	TRUE
40	device	'172.16.4.20'	'Telco North'	may_terminate	enrolled	'172.16.4.20'	h323-Q931	1.4.3	''	'11'	on-line	32	''	TRUE
41	device	'172.16.4.21'	'Telco North'	may_terminate	not_enrolle	'172.16.4.21'	h323-Q931	1.4.3	''	''	off-line	32	''	TRUE
42	device	'123.72.89.6'	'Telco South'	may_terminate	enrolled	'Gateway1'	h323-Q931	1.4.3	''	''	on-line	32	''	TRUE
43	device	'132.40.67.99'	'Telco South'	may_not_terminate	enrolled	'Gateway2'	h323-Q931	1.4.3	''	''	on-line	32	''	TRUE
44	term-devices	'Hong Kong'												
45	term-devices	'IXC London'	192.168.1.199	1										
46	term-devices	'IXC Los Angeles'	GW_B	1										
47	term-devices	'IXC New York'	GW_A	1										
48	term-devices	'Mexico'	208.57.120.64	1										
49	term-devices	'PTT VoIP Operations'												
50	term-devices	'Telco North POP'	172.16.4.21	1	172.16.4.2(	1								
51	term-devices	'Telco South NOC'	Gateway2	1										
52	route	'default'	'1212'	IXC New York	1	10000	PTT VoIP (	2	10000	Telco North	0	10000		
53	route	'default'	'1404'											
54	route	'default'	'1408'											

5. Dial Plan Assignment Configuration Files

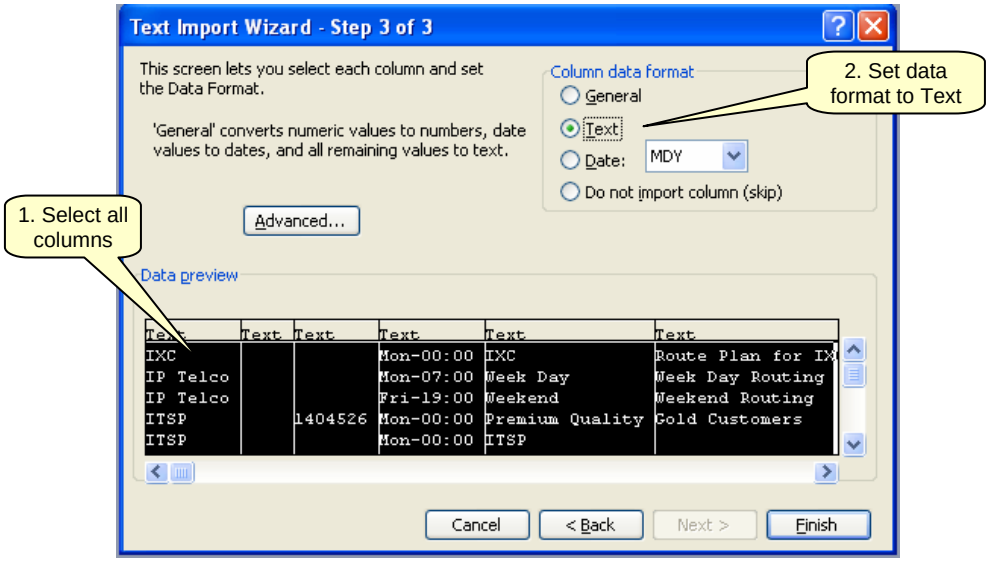
The NexSRS server allows up to four route plans to be assigned for each Customer. The Route Plan assignments are configured in four separate files:

1. `dial_plan_1.cfg`
2. `dial_plan_2.cfg`
3. `dial_plan_3.cfg`
4. `dial_plan_4.cfg`

The Dial Plan assignment configuration files are text files and can be created or edited manually using any text editor or Microsoft Excel. Below is an example Dial Plan file which has been imported to Excel. The column titles (row 1) have been added in Excel and are not part of the original file format.

	A	B	C	D	E	F
1	Customer	Called Number	Calling Number	Start Time	Product Assigned	Comment
2	IXC			Mon-00:00	IXC	Route Plan for IXC
3	IP Telco			Mon-07:00	Week Day	Week Day Routing
4	IP Telco			Fri-19:00	Weekend	Weekend Routing
5	ITSP		1404526	Mon-00:00	Premium Quality	Gold Customers
6	ITSP			Mon-00:00	ITSP	

Note: When importing a text from Excel, make certain to define all values are text. Otherwise Excel may perform formatting operations (such as convert a number to date) which change the original values. The screenshot below shows how to use the Excel text import wizard to set all value to text values.


Text Import Wizard - Step 3 of 3
This screen lets you select each column and set the Data Format.
'General' converts numeric values to numbers, date values to dates, and all remaining values to text.
Column data format
General
Text
Date:
MDY
Do not import column (skip)
2. Set data format to Text
Advanced...
1. Select all columns
Data preview
Text Text Text Text Text Text
IXC Mon-00:00 IXC Route Plan for IX
IP Telco Mon-07:00 Week Day Week Day Routing
IP Telco Fri-19:00 Weekend Weekend Routing
ITSP 1404526 Mon-00:00 Premium Quality Gold Customers
ITSP Mon-00:00 ITSP
Cancel
< Back
Next >
Finish

6. Number Translation Configuration Files

Manual Number Translation Configuration

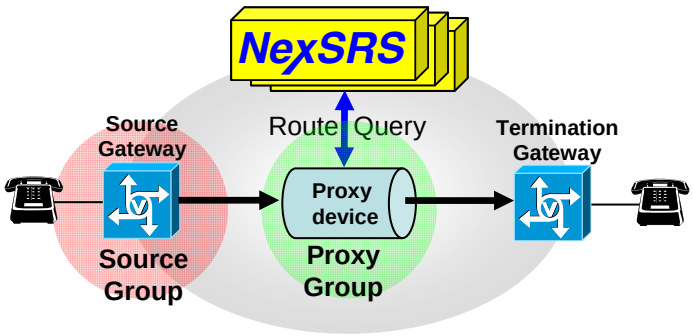
The number translation configuration files are text files which can be edited easily in Microsoft Excel of any other program which can edit text files. Below is a number translation configuration file which has been imported to Excel. The column titles, row 1, have been added in Excel and are not part of the original configuration file.

	A	B	C	D	E	F
1	Group	Called Number	Calling Number	Start Time	Translated Called Number	Comment
2	PTT		0900	Mon-00:00	Blocked	
3	PTT	0900		Mon-00:00	Blocked	
4	ITSP	18005266060		Mon-00:00	15402200309	After Hours Support
5	ITSP	18005266060		Mon-09:00	14045266052	Customer Support
6	Telco North	09		Mon-00:00	011+	Prefix Translation

Note: When importing a text file into Excel, make certain to define all numeric values as text. Otherwise, Excel will truncate numbers which begin with zero. For example, 09 will be converted to 9.

Number Translation Algorithm

The algorithms for pre-routing and post-routing number translation are identical. Called number translation policies are defined by Customer. The first step in the number translation algorithm is to determine which Customer's number translation policy applies for the call. The source device Customer has the highest priority, followed by the proxy device Customer, followed by the default Customer number translation policy which applies for all Customers. For example, in the call scenario illustrated below the number translation policy of the Source VoIP Device Customer has the highest priority.

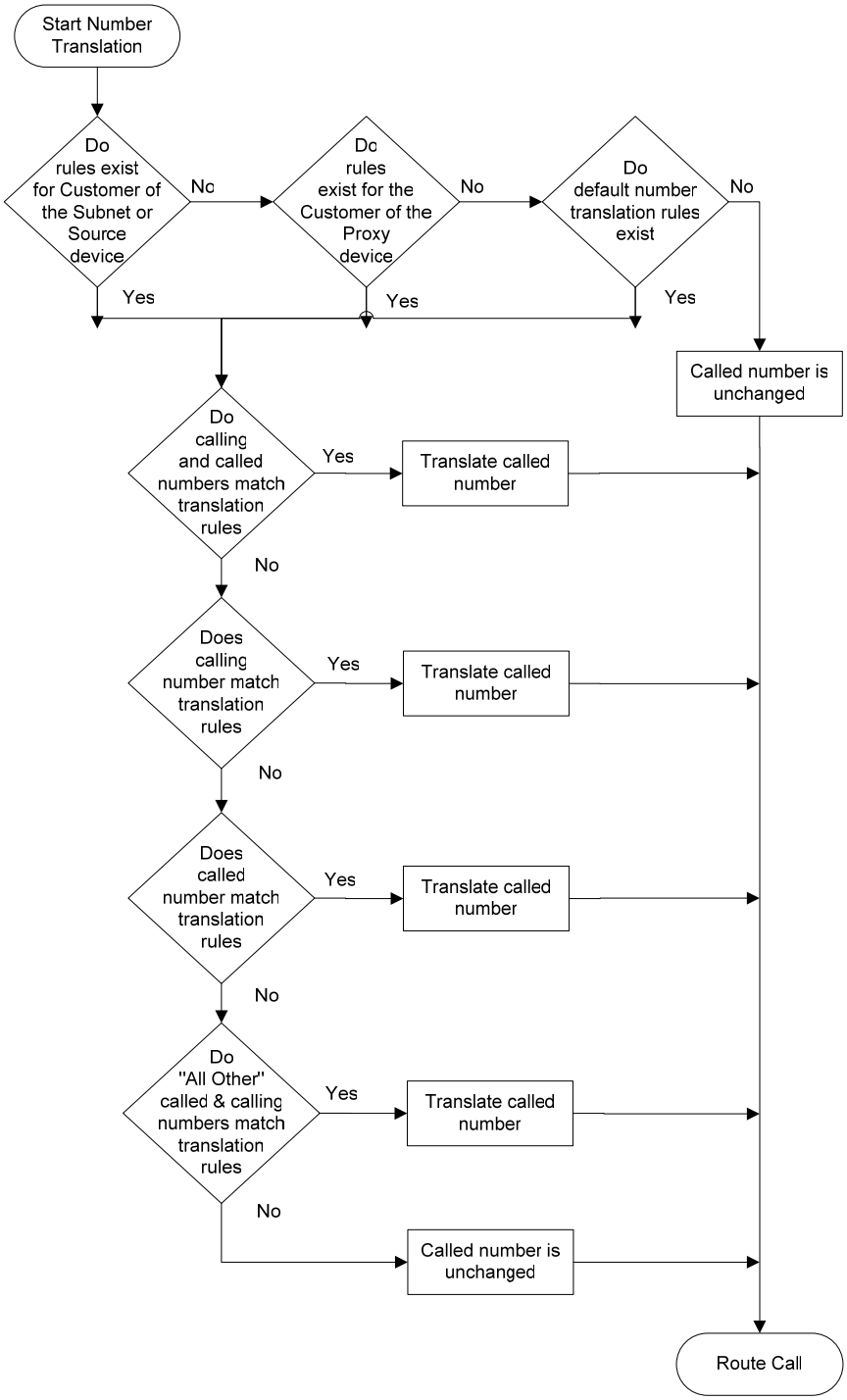


However, if no translation rules are defined for the Source VoIP Device Customer, the NexSRS server will use the number translation rules defined for the Proxy device Customer. If no number translation rules are defined for the Proxy device Customer, the NexSRS server will use the default number translation rules if they exist.

After determining which Customer's number translation rules to use, the next step in the algorithm is to iterate through the four following sets of number translation rules, in the order shown, to find a match between the called / calling numbers of the call setup from the Source VoIP Device and a number translation rule.

1. Translation rules based on the called and calling numbers.
2. Translation rules based on calling number.
3. Translation rules based on called number.
4. Translation rules based on all other calling and called numbers.

The first rule with a matching calling and/or called number is the rule used.



7. Called Number Digit Filtering Configuration File

The `pre_routing_called_number_digit_filter.cfg` file is a text file similar to the the number translation configuration files. The screenshot below is taken from a `pre_routing_called_number_digit_filter.cfg` file which has been imported to Microsoft Excel. This screenshot illustrates the data format for the file. The column headings (row 1) was added in Excel and are not part of the original file format.

	A	B	C	D	E	F
1	Customer	Called Number	Calling Number	Start Time	Called Number Length Range	Comment
2	PTT	1404		Mon-00:00	10-12	
3	IP Telco	1		Mon-00:00	11	Dial Filter for North America

8. Termination Cause Codes

The Termination Cause Code (TCCode) field in a Nexus CDR contains the call disconnect reason reported by the VoIP device. For most VoIP devices the reported TC Code will be an ISDN disconnect code. However, most SIP devices do not report ISDN disconnect codes and report the SIP error message code. The table below provides a list and description of TC Codes.

TC Code	Description	Type
1	unallocated (unassigned) number	ISDN
2	no route to specified transit network	ISDN
3	no route to destination	ISDN
4	send special information tone	ISDN
5	misdialed trunk prefix	ISDN
6	channel unacceptable	ISDN
7	call awarded and being delivered in an established channel	ISDN
8	preemption	ISDN
9	preemption – circuit reserved for re-use	ISDN
16	normal call clearing	ISDN
17	user busy	ISDN
18	no user responding	ISDN
19	no answer from user (user alerted)	ISDN
20	subscriber absent	ISDN
21	call rejected	ISDN
22	number changed	ISDN
26	non-selected user clearing	ISDN
27	destination out of order	ISDN
28	invalid number format (address incomplete)	ISDN
29	facility rejected	ISDN
30	response to STATUS ENQUIRY	ISDN
31	normal, unspecified	ISDN
32	no circuit/channel unavailable	OSP
34	no circuit/channel unavailable	ISDN
38	network out of order	ISDN
39	permanent frame mode connection out of service	ISDN
40	permanent frame mode connection operational	ISDN
41	temporary failure	ISDN
42	switching equipment congestion	ISDN
43	access information discarded	ISDN
44	requested circuit/channel not available	ISDN
46	precedence call blocked	ISDN
47	resource unavailable, unspecified	ISDN
49	quality of service unavailable	ISDN
50	requested facility not subscribed	ISDN
53	outgoing calls barred within closed user group	ISDN
55	incoming calls barred with closed user group	ISDN
57	bearer capability not authorized	ISDN

58	bearer capability not presently available	ISDN
62	inconsistency in designated outgoing access information and subscriber class	ISDN
63	service or option not available, unspecified	ISDN
65	bearer capability not implemented	ISDN
66	channel type not implemented	ISDN
69	requested facility not implemented	ISDN
70	only restricted digital information bearer capability is available	ISDN
79	service or option nit implemented, unspecified	ISDN
81	invalid call reference value	ISDN
82	identified channel does not exist	ISDN
83	a suspended call exists, but this call identity does not	ISDN
84	call identity in use	ISDN
85	no call suspended	ISDN
86	call having the requested call identity has been cleared	ISDN
87	user not member of closed user group	ISDN
88	incompatible destination	ISDN
90	non-existent closed user group	ISDN
91	invalid transit network selection	ISDN
95	invalid message, unspecified	ISDN
96	mandatory information element is missing	ISDN
97	message type non-existent or not implemented	ISDN
98	message not compatible with call state or message type non-existent or not implemented	ISDN
99	information element/parameter non-existent or not implemented	ISDN
100	invalid information element contents	ISDN
101	message not compatible with call state	ISDN
102	recovery on timer expiry	ISDN
103	parameter non-existent or not implemented, passed on	ISDN
110	message with unrecognized parameter, discarded	ISDN
111	protocol error, unspecified	ISDN
127	interworking, unspecified	ISDN
301	Moved permanently – maps to ISDN code 22	SIP
403	Forbidden – maps to ISDN code 21, 55, 57 and 87	SIP
404	Not Found – maps to ISDN codes 1, 2, 3 and 26	SIP
408	Request Timeout – maps to ISDN code 18	SIP
410	Gone – maps to ISDN codes 22 and 23	SIP
480	Temporarily unavailable – maps to ISDN codes 19, 20 and 31	SIP
484	Address incomplete – maps to ISDN code 28	SIP
486	Busy here – maps to ISDN code 17	SIP
488	Not Acceptable Here – maps to ISDN codes 65 and 70	SIP
500	Server internal error – maps to ISDN codes 111 and 127	SIP
501	Not implemented – maps to ISDN code 29 and 79	SIP
502	Bad Gateway – maps to ISDN code 27	SIP
503	Service unavailable – maps to ISDN codes 34, 38, 41, 42, 47, 58 and 58	SIP
504	Gateway timeout – maps to ISDN code 102	SIP

1016	normal call clearing	OSP
10016	TC Code in call start CDR from a stateless proxy indicating the call has connected and started	OSP

References:

ITU Q931 provides a listing and description of ISDN (ISUP) disconnect codes.

RFC 3666 SIP – PSTN Call Flows; Provides a mapping of SIP and ISUP codes.

9. inter_osp_config.sh

The inter_osp_config.sh file is shown below. This file is used to configure an enterprise NexSRS server for OSP ExtraNet queries to a carrier NexSRS server.

```
##########################################################################
#                                                                        #
#                           inter_osp_config.sh                          #
#                                                                        #
##########################################################################
#
# This is the configuration file for implementing OSP messages
# between an Enterprise NexSRS server and it's Inter-Domain Carrier.
#
# When an Enterprise NexSRS server cannot route a VoIP call because the
# route does not exist in its routing table, the Enterprise NexSRS
# server will automatically send an OSP AuthorizationRequest to the OSP
# Server of its Inter-Domain Carrier requesting a route to an external
# VoIP domain that can complete the call.
#
#  |<--- Enterprise VoIP Domain --->|<--- External VoIP Domains --->|
#
#            ============                        ===============
#           | Enterprise | ==== AuthReq ===>   | Inter-Domain  |
#           |   NexSRS   |                     |    Carrier    |
#           |   server   | <=== AuthRsp ====   |   OSP server  |
#            ============                        ===============
#
#
# Enterprise NexSRS Server Public IP Address
# ==========================================
# When the Enterprise NexSRS server sends an AuthorizationRequest to
# it's Inter-Domain Carrier it must identify itself by including its
# public IP address in the AuthorizationRequest message to the carrier
# NexSRS server.  If the Enterprise NexSRS server is behind a firewall,
# its machine IP address will be a private IP address.  This parameter
# replaces the private IP address Enterprise NexSRS server with its
# public IP address for communications with an external NexSRS server.
#
# If set to a public IP address, the address must be entered in
# brackets, for example, [123.123.123.123].  This parameter may also be
# be specified as a fully qualified domain name that can be resolved
# by any public Domain Name Server, for example, www.transnexus.com.
#
SERVER_PUBLIC_IP_ADDRESS="`hostname`.`domainname`"
export SERVER_PUBLIC_IP_ADDRESS

#
# The Enterprise NexSRS server uses ".rnd" file for seeding pseudo-
random
# number generator engine.
#

sum nexus.cfg > .rnd
```

10. Utility Scripts

This section contains a printout of the NexSRS utility scripts. For more information on these scripts, refer to the 'Utilities' section in Chapter 7 NexSRS Server Operation.

Compress Archived CDR Files

```
#!/usr/bin/sh
# compress_archived_cdr_files.sh
#
# This script uses the gzip utility to compress all files with a .cdr
# extension in the $NexSRS_HOME/xitami/osp_ui/cdrs/ARCHIVE directory.
# This script must be run by a user ospadmin.

if [ ! "$NexSRS_HOME" ]
then
        echo "The NexSRS_HOME variable is not set."
        exit 1
fi

if [ ! -d $NexSRS_HOME/utils ]
then
        echo "Directory $NexSRS_HOME/utils does not exist"
        echo "Make sure that NexSRS_HOME variable is correctly set."
        exit 1
fi

cd $NexSRS_HOME/utils

gzip ../xitami/osp_ui/cdrs/ARCHIVE/*.cdr
```

Compress Backup Files

```
#!/usr/bin/sh
# compress_backup_files.sh
#
# This script uses the gzip utility to compress all files with a .bu
# extension in the $NexSRS_HOME/BACKUP directory.
# This script must be run by a user ospadmin.

if [ ! "$NexSRS_HOME" ]
then
             echo "The NexSRS_HOME variable is not set."
             exit 1
fi

if [ ! -d $NexSRS_HOME/utils ]
then
             echo "Directory $NexSRS_HOME/utils does not exist"
             echo "Make sure that NexSRS_HOME variable is correctly set."
             exit 1
fi

cd $NexSRS_HOME/utils

gzip ../BACKUP/*.bu
```

Cut NexSRS Log File

```
#!/usr/bin/sh
# cut_nexus_log.sh
#
# The NexSRS log file, nexus.log, grows continuously and can become
# too large to view after several months of normal operation.  This
```

```
# script, when run periodically, solves this operational problem by
# renaming and saving the current log file and allowing the NexSRS
# server to start a new nexus.log file.
#
# When this file is run, the current nexus.log file is renamed and then
# gzipped.  The file name for the backup is nexus.log-YYYYMMDD-HH:MM.gz.
# Where YYYYMMDD-HH:MM stands for the year, month, day, hour and minute
# when the log file was backed up.  When the nexus.log file is renamed
# and backed up, a new nexus.log file is automatically written in the
# $NexSRS_HOME/xitiami/osp_ui/log directory.
#
# This script must be run by user ospadmin.

BACKUP_LOG_FILE_NAME=nexus.log-`date '+%Y%m%d-%H:%M'`

if [ ! "$NexSRS_HOME" ]
then
        echo "The NexSRS_HOME variable is not set."
        exit 1
fi

if [ ! -d $NexSRS_HOME/utils ]
then
        echo "Directory $NexSRS_HOME/utils does not exist"
        echo "Make sure that NexSRS_HOME variable is correctly set."
        exit 1
fi

cd $NexSRS_HOME/utils

cd ../xitami/osp_ui/log

mv nexus.log $BACKUP_LOG_FILE_NAME

gzip $BACKUP_LOG_FILE_NAME
```

Delete Old Archived NexSRS Log Files

```
#!/usr/bin/sh
# delete_archived_nexus_log_files.sh
#
# This script automates the process of deleting archived files from the
# $NexSRS_HOME/xitami/osp_ui/log directory that have a .gz extension and have
been
# unchanged for more days than the number of days defined by the
# DELETE_FILES_OLDER_THAN_DAYS variable.  The default value for
# this variable is 45 days.
#
# For example, if this script is run with DELETE_FILES_OLDER_THAN_DAYS=45
# all .gz files that have not been changed, compressed, renamed, etc. for
# more than 45 days will be deleted.
#
# This script can be run by user ospadmin.  This script uses the
# IsFileOldEnough.pl script which must also be located in the
# $NexSRS_HOME/utils directory.

DELETE_FILES_OLDER_THAN_DAYS=45
FILE_MASK=../xitami/osp_ui/log/*.gz*

if [ ! "$NexSRS_HOME" ]
then
        echo "The NexSRS_HOME variable is not set."
        exit 1
```

```
fi

if [ ! -d $NexSRS_HOME/utils ]
then
        echo "Directory $NexSRS_HOME/utils does not exist"
        echo "Make sure that NexSRS_HOME variable is correctly set."
        exit 1
fi

cd $NexSRS_HOME/utils

for file in `ls $FILE_MASK`
do
  if ./IsFileOldEnough.pl $DELETE_FILES_OLDER_THAN_DAYS $file
  then rm ./$file
  fi
done
```

Delete Old NexSRSBackup Files

```
#!/usr/bin/sh
# delete_old_backup_files.sh
#
# This script automates the process of deleting all files from the
# $NexSRS_HOME/BACKUP directory that have a .bu.gz extension and have been
# unchanged for more days than the number of days defined by the
# DELETE_FILES_OLDER_THAN_DAYS variable.  The default value for
# this variable is 45 days.
#
# For example, if this script is run with DELETE_FILES_OLDER_THAN_DAYS=45
# all .gz files, compressed and uncompressed, that have not been changed,
# compressed, renamed, etc. for more than 45 days will be deleted.
#
# This script can be run by user ospadmin.  This script uses the
# IsFileOldEnough.pl script which must also be located in the
# $NexSRS_HOME/utils directory.

DELETE_FILES_OLDER_THAN_DAYS=45
FILE_MASK=../BACKUP/*.bu.gz

if [ ! "$NexSRS_HOME" ]
then
        echo "The NexSRS_HOME variable is not set."
        exit 1
fi

if [ ! -d $NexSRS_HOME/utils ]
then
        echo "Directory $NexSRS_HOME/utils does not exist"
        echo "Make sure that NexSRS_HOME variable is correctly set."
        exit 1
fi

cd $NexSRS_HOME/utils

for file in `ls $FILE_MASK`
do
  if ./IsFileOldEnough.pl $DELETE_FILES_OLDER_THAN_DAYS $file
  then rm ./$file
  fi
done
```

Delete Old CDR Files

```
#!/usr/bin/sh
# delete_old_cdr_files.sh
#
# This script automates the process of deleting all files from the
# $NexSRS_HOME/xitami/osp_ui/cdrs/ARCHIVE directory that have a .cdr extension
# and have been unchanged for more days than the number of days
# defined by the DELETE_FILES_OLDER_THAN_DAYS variable.  The default
# value for this variable is 45 days.
#
# For example, DELETE_FILES_OLDER_THAN_DAYS=45 all .cdr files,
# compressed and uncompressed, that have not been changed,
# compressed, renamed, etc. for more than 45 days will be deleted.
#
# This script must be run by user ospadmin.  This script uses the
# IsFileOldEnough.pl script which must also be located in the
# $NexSRS_HOME/utils directory.

DELETE_FILES_OLDER_THAN_DAYS=45
FILE_MASK=../xitami/osp_ui/cdrs/ARCHIVE/*.cdr*

if [ ! "$NexSRS_HOME" ]
then
        echo "The NexSRS_HOME variable is not set."
        exit 1
fi

if [ ! -d $NexSRS_HOME/utils ]
then
        echo "Directory $NexSRS_HOME/utils does not exist"
        echo "Make sure that NexSRS_HOME variable is correctly set."
        exit 1
fi

cd $NexSRS_HOME/utils

for file in `ls $FILE_MASK`
do
  if ./IsFileOldEnough.pl $DELETE_FILES_OLDER_THAN_DAYS $file
  then rm ./$file
  fi
done
```

11. SNMP_config.sh

The NexSRS server includes a SNMP module. The configuration file for the SNMP module is provided below.

```
#########################################################################
#
#                              snmp_config.sh
#
#########################################################################
# This file is the configuration file for Nexus WatchDog(TM) SNMP
# application which monitors the status of the OSP Nexus Server.  The
# Nexus WatchDog supports the MIB-II system Group.  The Objects in the
# system Group provide the following overall information about the OSP
# Nexus Server.
#
# MIB II system Group Objects:
# ============================
# sysDescr    -(hardcoded) "Nexus WatchDog Version: M.mm Build:YYYYMMDD"
# sysObjectID -(hardcoded) "1.3.6.1.4.1.3241.1.1"
# sysUpTime   -(calculated) Time since starting application
# sysContact  -(configured) see below: SNMP_SYSTEM_GROUP_CONTACT
# sysName     -(configured) see below: SNMP_SYSTEM_GROUP_NAME
# sysLocation -(configured) see below: SNMP_SYSTEM_GROUP_LOCATION
# sysServices -(hardcoded)  64  (application layer 7 in OSI model)
#
# Traps:
# ======
# The Nexus WatchDog also uses the following generic traps to alert an
# SNMP manager of changes in operating status:
#
# - coldStart(0)
#   This trap is sent to SNMP_TRAPS_HOST when the Nexus WatchDog
#   re-starts either the web server or OSP Nexus Server.  A coldStart
#   trap is not sent when the web server or OSP Nexus Server is started
#   for the first time.
#
# - warmStart(1)
#   This trap is sent to SNMP_TRAPS_HOST when the Nexus WatchDog
#   starts.
#

#                    CONFIGURABLE PARAMETERS
#                    =======================
# The following is the list of a configurable parameters with a
# description for each.  Change these parameters to customize your
# Nexus WatchDog SNMP configuration.
#
#
#                         SNMP_ENABLE
#
# This boolean variable may be either Yes or No.
# If SNMP_ENABLE=Yes then SNMP application is enabled.
# IF SNMP_ENABLE=No then SNMP application is disabled.
# The default value is No
#
```

```
SNMP_ENABLE=No

#
#                       SNMP_AGENT_PORT
#
# If SNMP is enabled, then Nexus WatchDog will listen on this port for
# messages from an SNMP manager to get or set MIB data.  The value of
# this variable may be any integer between 0 and 65,535.  The default
# value of this variable is 161, the default TCP-IP port for SNMP
# agents. However, the Nexus WatchDog has been pre-configured with
# SNMP_AGENT_PORT=1161 to avoid any potential conflict with the Solaris
# operating system which uses port 161 for its own SNMP agent.
#
SNMP_AGENT_PORT=1161

#
#                SNMP_SYSTEM_GROUP Parameters
#
# The following three parameters define objects which are specified by
# the MIB II system Group.  They are listed below with their default
# values.
# SNMP_SYSTEM_GROUP_CONTACT="support@transnexus.com"
# SNMP_SYSTEM_GROUP_NAME="OSP Nexus Server"
# SNMP_SYSTEM_GROUP_LOCATION=`uname -a`
# The values of these parameters must be string variables limited to
# 100 characters.  The string value should be enclosed with quotation
# marks if spaces are included in the string.  The value `uname a` uses
# this Unix command to populate the string value with the local host
# name.
#
SNMP_SYSTEM_GROUP_CONTACT="support@transnexus.com"
SNMP_SYSTEM_GROUP_NAME="OSP Nexus Server"
SNMP_SYSTEM_GROUP_LOCATION=`uname -a`

#
#                       SNMP_TRAPS_ENABLE
#
# This boolean value may be set to Yes or No.  The default value is No.
# If SNMP_TRAPS_ENABLE=No, traps are disabled.
# If SNMP_TRAPS_ENABLE=Yes, traps are enabled.
#
SNMP_TRAPS_ENABLE=No

#
#                        SNMP_TRAPS_HOST
#
# This parameter defines the device name or IP address of the SNMP
# manager to which the Nexus WatchDog will send traps.  This variable
# must be a string less than 100 characters.  The default is localhost.
#
SNMP_TRAPS_HOST="localhost"

#
#                        SNMP_TRAPS_PORT
#
# If SNMP is enabled, the Nexus WatchDog will send traps to this port
# on the SNMP_TRAPS_HOST.  This parameter may be any integer value
```

```
# between 0 and 65,535.  The default value is 162, the default port
# used by SNMP managers to receive trap messages.
#
SNMP_TRAPS_PORT=162

#
#                    COMMUNITY AUTHENTICATION
#
# The Nexus WatchDog uses password authentication procedures defined in
# in RFC 1157 for authenticating messages from SMNP managers.  Every
# SNMP get or put request from an SNMP manager to the Nexus WatchDog
# includes a community name which the Nexus WatchDog uses to
# authenticate the SNMP manager request.
#
# The SNMP_READ_COMMUNITY="public" parameter defines public as the
# community name required get data from the SNMP agent MIB.
#
# The SNMP_WRITE_COMMUNITY="public" parameter defines public as the
# community name required to put or write data to the SNMP agent MIB.
#
# Both of these parameters must be string values of 100 characters or
# less.  These variables are not configurable in this release and are
# always set the default value of public.

# Other Information:
# ==================
# The Nexus WatchDog SNMP application uses the following OIDs.
# TransNexus  OID: 1.3.6.1.4.1.3241
# NexusServer OID: 1.3.6.1.4.1.3241.1
# WatchDog    OID: 1.3.6.1.4.1.3241.1.1
#
# This script, is run by the start_osp_server.sh script every time the
# OSP Nexus Server is restarted.  To change SNMP settings for the Nexus
# WatchDog application, the OSP Nexus Server must be shutdown and
# restarted with the start_osp_server.sh script.  Running this script
# alone will not change SNMP settings.
#

export SNMP_ENABLE
export SNMP_AGENT_PORT
export SNMP_TRAPS_ENABLE
export SNMP_TRAPS_HOST
export SNMP_TRAPS_PORT
export SNMP_SYSTEM_GROUP_CONTACT
export SNMP_SYSTEM_GROUP_NAME
export SNMP_SYSTEM_GROUP_LOCATION
```

Index

400 Error Code 94
401 Error Code 95
404 Error Codes 97
503 Error Codes 98
Add New Device 25
Adding Subnets 22
AUTHENTICATE_ACTUAL_SOURCE_DEVICE 9, 102
Authorization and Routing 20
Breakouts 30
Busy Hour Call Attempts 16
CALEA 47
Call Detail Collection 20
Call Detail Records 63
Called Number 70, 71
Callid 69
CDR_FILE_TIME_SPAN 9
certificate 7
CHECK_FOR_DUPLICATE_DESTINATIONS 10
CRON 60
CRYPTO_HW_ACCELERATION 10
cryptographic acceleration 4
Destination Protocol 24, 26
Destinations 30
Device Configuration 24
Device Enrollment ..... *See* log messages
Dial Peers 109
Dial Plans 39
DISK_SPACE_WARNING 11
ENGINE_ID_NUM 8, 9
ENUM 47
ExtraNet 101, 135
ExtraNet Routing 44
Features 20
Groups 21
Inter Domain Carrier 17, 46
inter_osp_config.sh 135
Interoperability 109
IP Masking 131
License ..... *See* Log messages
license key 16, 19
License Upgrade 18
Load Balancing 34
Load Balancing Algorithm 121
Log messages 80
Log Messages 56
LookAhead Routing 24, 26
Maximum Devices 16
Maximum Number of Digits 16
MAXIMUM_CALL_LENGTH 11
nexus.cfg 57, 123, 124
nexus_cfg.bu 104
Number Translation 48, 127, 128
Number Translation Algorithm 128
Operating Scripts 55
OSP Client Errors 99
OSP Version 24, 26
Platform Requirements 4
Prefix Translation 52
Products 32
Questions and Answers 104
Rank Ordering 34
Release Notes 114
Routes 21
Routing Table 124
SCEP 89
settlement 7
Short Name 23, 25
shut down 8
SSL 7
Subnets 131
Suggested Routes 44, 113
Tech Prefix Inbound 24, 26
TEP 89
Termination Cause Codes 69, 132
top 105
Transaction ID 69
Upgrade Procedures 12
User Administration 74
user interface 19, 25
UTC 107
Utilities 59
VoIP Domain ID 17
VoIP Domain Name 17
Watch Dog 62
Xitami 7, 74